SKELETONIZATION-BASED AUTOMATED TRACING AND RECONSTRUCTION OF NEUROVASCULAR NETWORKS IN KNIFE-EDGE SCANNING MICROSCOPE MOUSE BRAIN INDIA INK DATASET

A Thesis

by

ANKUR SINGHAL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Yoonsuck Choe
Committee Members,	John Keyser
	Louise Abbott
Head of Department,	Dilma Da Silva

August 2015

Major Subject: Computer Science

ABSTRACT

The vascular architecture of the brain is very complex and reconstruction of this architecture can aid in understanding the functions of vessels in different regions of the brain. Advances in microscopy have enabled high-throughput imaging of massive volumes of biological microstructure at a very high resolution. The Knife Edge Scanning Microscope (KESM), developed by the Brain Network Laboratory at Texas A & M University, is one such instrument that enables imaging of whole small animal brains at sub-micrometer resolution. The KESM has been successfully used to acquire vasculature dataset from a mouse brain stained by India ink. However, manual tracing and reconstruction of vessels is not feasible due to the huge volume of the dataset. Therefore, developing efficient and robust automatic tracing methods is essential for analysis of the network.

This thesis presents an efficient skeletonization based tracing algorithm to reconstruct the vascular structure in the KESM India ink data set. The skeleton is generated from the original volume by repetitively removing voxels from the object's boundary such that the connectivity and topology in the original volume is preserved. The skeleton is then dilated to reconstruct the volume. The accuracy of this method is determined by comparing the reconstructed volume with the original volume. This method is expected to trace the entire vascular network of the brain quickly and with high accuracy without human assistance.

DEDICATION

To my family.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Yoonsuck Choe for his constant support, guidance, and encouragement throughout the course of this research. I would also like to thank my committee members, Dr. John Keyser and Dr. Louise Abbott for their guidance and valuable comments on my research. I am very grateful to my friends and colleagues and the department faculty and staff for their support during my research work and for making my time at Texas A&M University a great experience. I would specially like to thank my lab members Wenjie Yang, Ananth Dileepkumar, Shashwat Lal Das and Manisha Srivastava for all the help they provided to me.

Finally, I would like to thank my family, especially my mother and father for all the love and support.

This research was funded in part by NSF grants #0905041 and #1208174. Data used in this project was enabled by NSF grants #0079874 and #1256086 and NIH/NINDS grant #1R01-NS54252.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGEMENTS iv

TABLE OF CONTENTS v

LIST OF FIGURES vii

LIST OF TABLES ix

1. INTRODUCTION 1

    1.1 Motivation 1
    1.2 Goal of the research 2
    1.3 Approach 2
    1.4 Significance 3
    1.5 Outline of the thesis 4

2. BACKGROUND AND RELATED WORK 5

    2.1 Background 5
    2.2 Related work 10

3. METHODS 17

    3.1 Generation of 3D volume from 2D image sequence 17
    3.2 Image preprocessing 17
    3.3 Skeletonization 19
    3.4 Reconstruction of the original volume 26

4. RESULTS AND ANALYSIS 30

    4.1 Results of skeleton generation 30
    4.2 Results of volume reconstruction 39
    4.3 Analysis of tracing speed 50

Page

5. DISCUSSION 53

5.1 Contribution 53
5.2 Open issues and future work 54

6. CONCLUSION 57

REFERENCES 58

LIST OF FIGURES

FIGURE		Page
2.1	Different parts of KESM	6
2.2	KESM cutting session	7
2.3	Imaging principle	8
2.4	Visualization of the vascular networks in the KESM India-Ink data	9
2.5	Overview of predictor-corrector algorithm	11
2.6	Illustration of MW algorithm for 2D tracing	13
2.7	Illustration of MIP based 3D vector tracing	13
2.8	“Moving flat box” for 3D tracing	15
3.1	Visualization of a 3D data volume in Paraview	18
3.2	Image preprocessing	19
3.3	Illustration of skeletonization of an object	20
3.4	Skeleton of a rectangle	21
3.5	Illustration of a non-simple boundary voxel	22
3.6	Indices of the 26-neighborhood of point v	22
3.7	Illustration of the necessity to perform rechecking	25
3.8	Overlay of original volume and its skeleton using Paraview	26
3.9	Visual comparison of original and reconstructed volume	28
4.1	The vascular tracing results of KESM 100×100×100 volume	31
4.2	The vascular tracing results of KESM 128×128×128 volume	32

FIGURE Page

4.3 The vascular tracing results of KESM 200×200×200 volume..... 33

4.4 The vascular tracing results of KESM 256×256×256 volume..... 34

4.5 View along x-axis for tracing results of KESM 512×512×512 volume..... 35

4.6 View along y-axis for tracing results of KESM 512×512×512 volume..... 35

4.7 View along z-axis for tracing results of KESM 512×512×512 volume..... 36

4.8 Skeleton size versus volume size 38

4.9 Number of branch points versus skeleton size..... 38

4.10 Original (left) and reconstructed (right) volumes of size 100×100×100.... 40

4.11 Original and reconstructed volumes of size 128×128×128 41

4.12 Original and reconstructed volumes of size 200×200×200 42

4.13 Original and reconstructed volumes of size 256×256×256 43

4.14 Original and reconstructed volumes of size 512×512×512 44

4.15 Quantitative analysis of the reconstructed volume 46

4.16 Histogram of diameter for volume of size 100×100×100 47

4.17 Histogram of diameter for volume of size 128×128×128 47

4.18 Histogram of diameter for volume of size 200×200×200 48

4.19 Histogram of diameter for volume of size 256×256×256 48

4.20 Histogram of diameter for volume of size 512×512×512 49

4.21 Processing time (in seconds) of skeletonization based tracing method 51

4.22 Processing time comparison 52

5.1 Illustration of imaging artifact and the resulting error in the skeleton..... 55

LIST OF TABLES

TABLE Page

4.1 Statistics collected for volumes of different sizes 37

4.2 Quantitative analysis of the reconstructed volumes 45

4.3 KL divergence between different histograms 50

1. INTRODUCTION

1.1 Motivation

The vascular architecture of the brain is very complex, and understanding of this architecture can be useful in understanding brain function and disorders [1]. Many medical conditions and disorders such as Alzheimer's disease and Parkinson's disease have neurovascular causes [2] and understanding structural abnormalities in brain microvasculature can be helpful in diagnosis and treatment of such disorders. Reconstruction and analysis of different structures in the brain can provide a very good understanding of their function in larger units. Advances in microscopy have enabled high-throughput imaging of massive volumes of biological microstructure at a very high resolution [3]. The Knife Edge Scanning Microscope (KESM), developed by the Brain Network Laboratory at Texas A & M University, is one such instrument which enables the imaging of whole small animal brains at sub-micrometer resolution (http://kesm.org).

The KESM has been successfully used to acquire vasculature dataset from a mouse brain stained by India ink [4]. Tracing and reconstruction of the entire vascular network is necessary to understand the architecture of the brain. Manual tracing is not feasible due to the huge volume of the dataset. Also, the dataset acquired from KESM might contain gaps and noise. Manual tracing in such large noisy datasets is very time consuming and not very accurate. Therefore, developing efficient and robust automatic tracing methods that can fully and accurately trace the entire network is essential for analysis of the network.

1.2 Goal of the research

The goal of this research is to develop an efficient skeletonization based vascular tracing algorithm that can be used to accurately trace the vessels in different regions of the brain and to reconstruct the neurovascular structure in the KESM India ink data set. The algorithm should be automatic and scalable to trace a large volume of the dataset. The algorithm should also be very fast so that the tracing results can be obtained quickly. The volume of the reconstructed vascular structure would be compared with the original volume to determine the accuracy of the approach.

1.3 Approach

Automatic tracing and reconstruction of the vascular structure in the dataset involved the following steps: generation of 3D volume from 2D image sequence, image preprocessing, skeleton generation from the volume, obtaining a VTK trace from the skeleton, dilation of the skeleton to reconstruct the original volume and quantitative analysis of the results.

First, the 2D images obtained from KESM were stacked together to generate a 3D volume. Next preprocessing was done on this volume. This involved filtering using Weiner filter [5] and thresholding using Ostu's method [6]. The filtering step removed the noise in the images and the thresholding step generated binary images.

Then this volume was used to generate the skeleton of the neurovascular structure using a thinning based 3D skeletonization algorithm [7]. The skeleton of an object is the locus of the centers of all inscribed maximal spheres where these spheres touch the object's boundary at more than one point [8]. The skeleton was obtained by

repetitively removing the voxels at the boundaries of these spheres which satisfied certain topological constraints. This skeleton was used to generate a VTK trace. This VTK trace was visually analyzed by overlaying the trace and the original volume using image processing software Paraview. Statistical information about the volume such as the number of nodes, number of branch termination points and the number of voxels in the skeleton also were calculated during this step.

Validation was done by reconstruction of the original volume using the skeleton. At each voxel in the skeleton, the vessel diameter was estimated from the original volume and the skeleton was dilated at that voxel using a sphere of the estimated diameter. Then the accuracy, precision and recall were calculated from the original and reconstructed volume. Also, the processing time of our method was observed and analyzed for volumes of different sizes.

1.4 Significance

This thesis presents an efficient skeletonization based tracing algorithm that can be used to reconstruct the vascular structure of the mouse brain dataset. The method presented is very accurate and can trace the entire network without human assistance. The method can handle vessels of varying thickness and is robust with respect to gaps and noise in the dataset. The method is also fast and can be used to obtain the results quickly even in large datasets. This method is expected to significantly contribute to the ongoing research into analysis of the vascular network of the brain.

1.5 Outline of the thesis

The thesis is organized as follows. In chapter 2, an overview of KESM and India ink dataset is provided and the related work done in this field is briefly summarized. In chapter 3, the algorithm used for vascular tracing is presented and the methodology of the thesis is discussed. In chapter 4, experimental results and analysis are presented. In chapter 5, the contributions, open issues and possible future work are discussed and in chapter 6, the conclusion is presented.

2. BACKGROUND AND RELATED WORK

In this chapter, the background and related work done in this area is presented. First, an overview of the KESM and the India ink stained mouse brain dataset is provided. Next, the different methods that have been explored for the tracing and reconstruction of neurovascular networks, are briefly summarized.

2.1 Background

2.1.1 Knife Edge Scanning Microscope

Reconstruction and analysis of the structure and interconnections of blood vessels is necessary for understanding their function in a broader perspective. To reconstruct and understand the vascular structure, it is necessary to first obtain images of the entire brain at a sub-micrometer resolution. The brain needs to be cut into sections and imaged at the same time to obtain high-throughput and high resolution volumetric datasets. The knife-edge scanning microscope (KESM), developed at the Brain Networks Laboratory (BNL), allows reconstruction of mammalian brain architecture quickly and accurately. The KESM can generate data at the rate of 180 MB per second. So, digital images of the entire mouse brain of size around $1cm^3$ can be obtained in less than 100 hours [9].

The different parts of the KESM are shown in figure 2.1. It consists of a knife-collimator assembly, a microscope, a high sensitivity line scan camera and a stacked series of mechanical stages. KESM allows us to cut and image the specimen

simultaneously. Thus, it acts as both, a microtome and a microscope [4]. Figure 2.2 shows a close-up view of KESM.

The knife-collimator assembly, consisting of a diamond knife and illuminator, provides illumination and a means to cut the specimen. The diamond knife is mounted in such a way that the top knife facet is 45° to the vertical and perpendicular to the axis of

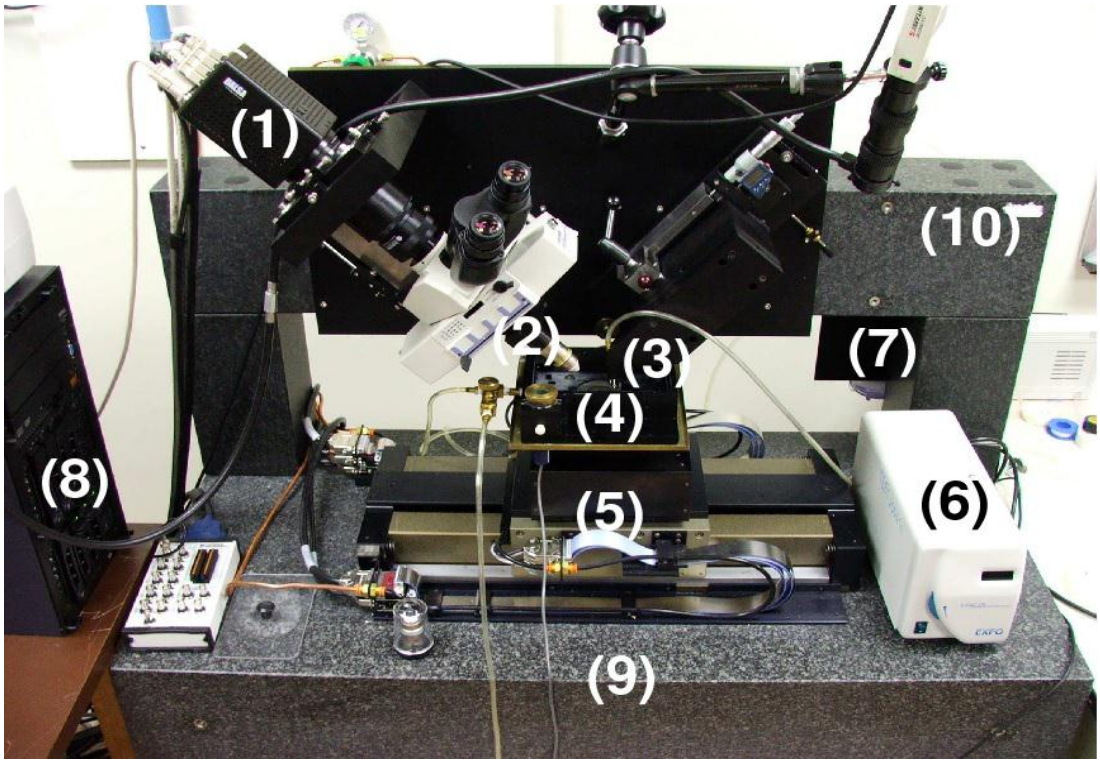


Figure 2.1: Different parts of KESM. (1) high-speed line scan camera, (2) microscope objective, (3) collimator assembly, (4) specimen tank, (5) air bearing three axis precision stage, (6) illuminator, (7) water pump, (8) computer server, (9) granite base and (10) granite bridge. Reprinted with permission from [10], © 2011 IEEE

the microscope. The mouse brain tissue, to be sectioned and imaged, is embedded in a plastic block and mounted on the specimen tank immersed under water. The diamond knife is kept stationary while the specimen tank is moved against the knife to cut the tissue. This motion is controlled by a series of mechanical and air-bearing stages. The generated sections are about 1μm thick [1].

The illuminator acts as a light source and the light is passed through the diamond knife to illuminate the tissue piece on the knife tip. The high-sensitivity line-scan camera scans the tissue through the microscope objective and sends the data directly to the

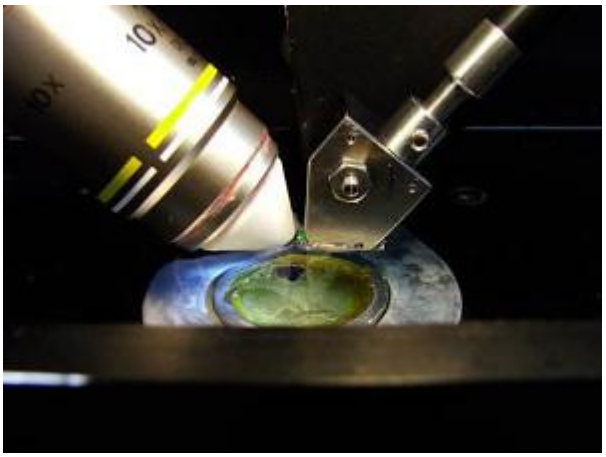


Figure 2.2: KESM cutting session [9]

camera server. In this way, we obtain a large stack of 2D images which can be converted to a 3D volume. Figure 2.3 shows the imaging principle of KESM.

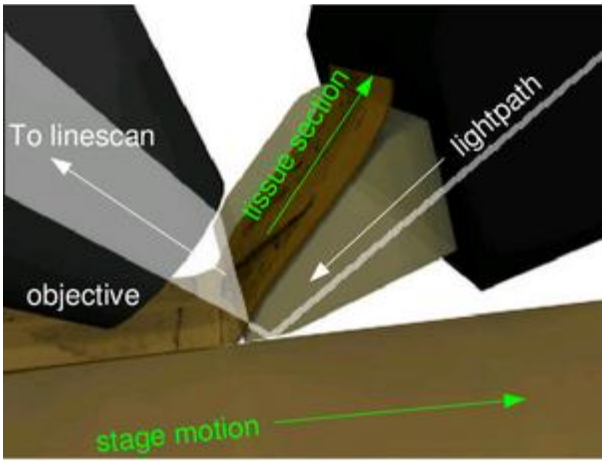


Figure 2.3: Imaging principle. Reprinted with permission from [10], © 2011 IEEE

2.1.2 KESM Data

Using the above procedure, we successfully obtained image data from mouse brains stained with different ink. Specifically, we obtained datasets from mouse brain stained in Golgi, Nissl and India ink. The Golgi and India ink datasets were obtained in 2008 while the Nissl dataset was obtained in 2010. Golgi dataset highlights the neuronal morphology in the brain while the Nissl dataset reveals the distribution of cell bodies [10].

In my thesis, we have worked on a dataset obtained from mouse brain stained with India ink. In this dataset, the vascular architecture of the brain is visible [10]. Since this is a digital data that was already available in the BNL, my research does not involve experimentation on live animals.

Figure 2.4 shows the visualization of the vascular networks in the dataset. In this figure, a lightly thresholded version and three fully thresholded versions along different

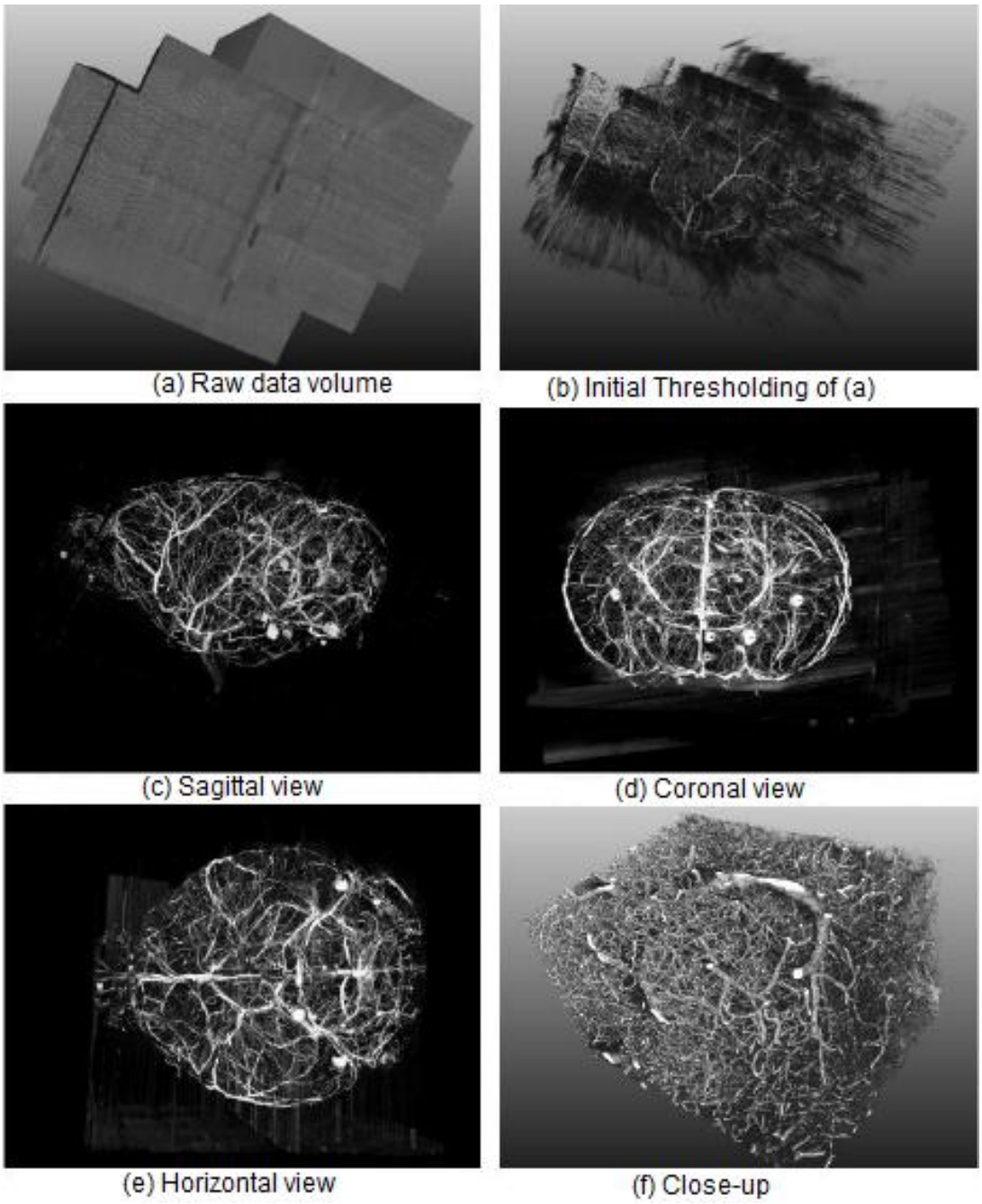


Figure 2.4 Visualization of the vascular networks in the KESM India-Ink data. (a) A raw data block in a sagittal view. (b) A lightly thresholded version of (a). (c)-(e) Fully thresholded versions of (a) but in different views (sagittal, coronal and horizontal). (f) Close-up of the intricate details within a 1.5 mm wide block. Reprinted with permission from [10], © 2011 IEEE

views (sagittal, coronal and horizontal) of a raw data block are shown.

The KESM datasets are basically stacks of 2 dimensional greyscale image sequences. These images are preprocessed and stacked together to provide a 3 dimensional view in Figure 2.4. The 3D view makes it easier to visualize and understand the structure of the volume. There are many 3D visualization software available. In my research, we have used Paraview to view the 3D volumes.

2.2 Related work

Mayerich et al. [11] presented a framework for segmenting filament networks from scalar volume using a predictor-corrector algorithm. They estimated the centerline of a filament by determining the path of a single particle, referred to as a tracer, over time, as it moved down each filament. Given a tracer state at time t, they used the predictor-corrector algorithm to compute the next tracer state at time (t+1). A tracer state consisted of three properties, tracer position p_t on the filament centerline, a vector v_t representing the estimated directory and a radius r_t defining the size of a template used to match filament cross-section. The predictor algorithm predicted a new trajectory v' and used it to compute new tracer position p', such that the new tracer position laid close to the centerline. The optimal vector v' was chosen from a set of vectors V = [v0, v1...vn] that was most closely aligned with the trajectory of the filament centerline. This was done by comparing a sample volume, starting at p_t and oriented such that the z-axis of the volume was aligned with the vector $v_i \in V$, with a template function and choosing the v_i that minimized a cost function. The new predicted position p' generally contained some error due to finite number of vectors in V used for prediction. This was corrected

by using a corrector algorithm that tested a set of points P = [p1,p2…pm] that laid in the plane defined by p' and close to p' and chose the point that minimized another cost function. This approach is significantly more efficient than template matching. However, creating sample volumes for prediction, correction and resizing requires a small region of data to be reconstructed and re-sampled. This is time consuming and computationally expensive. Figure 2.5 shows an overview of predictor-corrector algorithm.

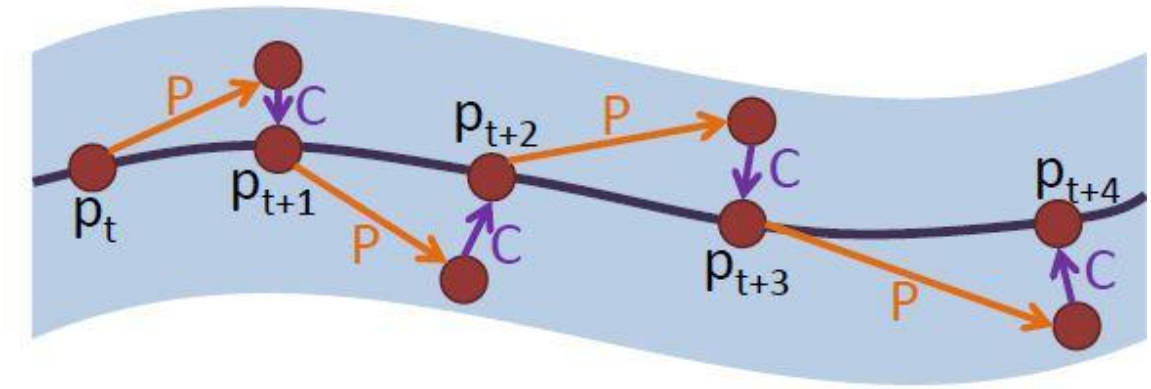


Figure 2.5: Overview of predictor-corrector algorithm. Predictor algorithm makes an initial prediction and advances the tracer. Correction algorithm further refines this prediction and makes the new particle position lie close to the centerline. Reprinted with permission from [11], © 2009 IEEE

Han et al. [12] developed a tracing algorithm based on Maximum Intensity Projection (MIP). First, they explained a method for tracing in 2D using a Moving Window (MW) algorithm. This gave a set of trace points along the medial axis of the fiber. These set of trace points were then connected using "Cubic Tangential Trace

Spline" (CTTS) for computing the medial axis. Figure 2.6 shows an illustration of MW algorithm. Next, using the 2D tracing algorithm, they described a Maximum Intensity Projection (MIP) based 3D vector tracing algorithm. MIP is a method that projects the maximum intensity value along each line perpendicular to the projected plane. First, three local MIPs were projected on 2D planes along each axis. Then, in each tracing step, the local MIP along the longest axis was ignored, as it contained less valuable information, and the other two MIPs were used to get the fiber directions, using any 2D tracing algorithm. These vectors were then combined to obtain the 3D fiber direction on the current voxel. Using this direction, the next candidate center was calculated. Figure 2.7 shows an example of MIP based 3D vector tracing. This exploration method is semi-automatic as the user has to provide an initial seed point to start the exploration of each branch. In a large dataset, this would require significant human effort and would be time-consuming. Also, it is assumed that one of the three projections will have less valuable information compared to the other two projections and it is ignored. If the vessel is at 45° to all the three axis, all the projections will have equal information about vessel direction and we will ignore useful information.

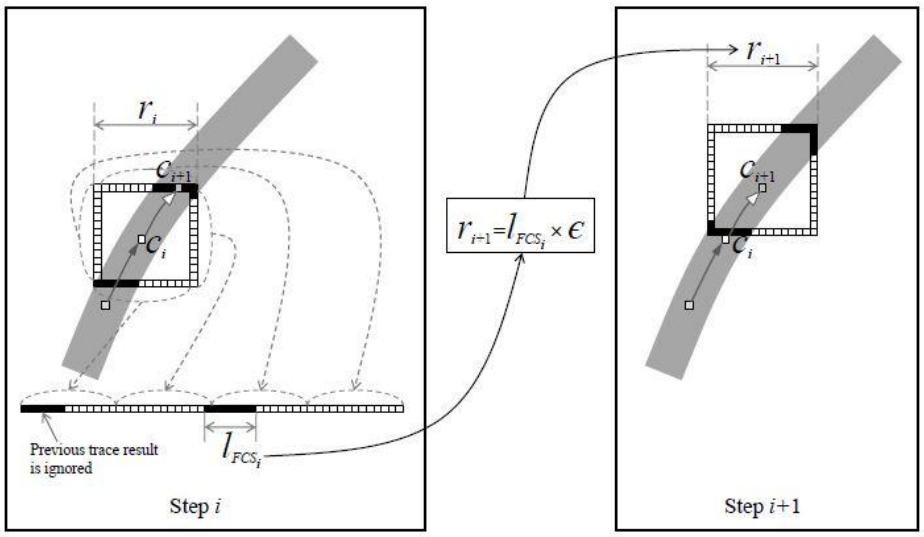


Figure 2.6: Illustration of MW algorithm for 2D tracing. FCS is fiber cross section. r_i is the side length of the i^{th} step's MW and c_i is the center of the i^{th} step's MV. l_{FCSi} is the length of the fiber cross section in i^{th} step. ϵ is the scaling factor and it is used to calculate r_{i+1}, the side length of the MW in $(i+1)^{th}$ step. c_{i+1} which is the center of l_{FCSi} is used the center of $(i+1)^{th}$ step's MW [12].

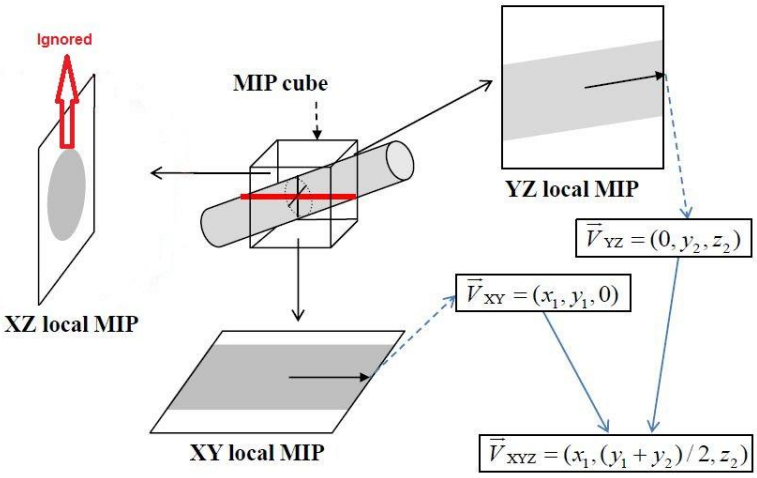


Figure 2.7: Illustration of MIP based 3D vector tracing. The local MIP along XZ plane is ignored as it doesn't contain much useful information for tracing. From the YZ and XY local MIP, 2D tracing directions are obtained and these 2D tracing directions are combined to form 3D tracing direction. Adapted from [12]

Yang et al. [13] described a tracing method using a “moving flat box”. The height of a flat box is much smaller than its length or width. In this approach, first, a seed point and an initial tracing direction were manually selected and the flat box was kept to let the seed point be the center of one of the surfaces of the flat box. The position of the other surface (predicting surface) depended on the initial tracing direction. Next, the vessel cross sectional regions on the predicting surface were calculated and using connected component labeling, centroids of the vessel cross section area on the predicting surface were found. The centroid of the cross section was considered to be the next candidate point and the moving trajectory of the vessel was obtained by connecting the seed point with the next candidate point. The next candidate point was used as the seed point in the next tracing step. Using a flat box significantly increases the processing speed compared to MIP because in the flat box method, only the top and bottom surfaces of the flat box are processed to trace the centerline of the vessel while the MIP method requires six surfaces of the MIP cubes to be processed in order to predict the vessel direction in 3D. However, this method requires initial seed points on the six surfaces of the volume and while tracing the volume, the same part of the vessel might be traced multiple times if multiple seed points on different surfaces lead to the same vessel, resulting in a significant increase in processing time. Also, the resulting trace is not very smooth as this method connects extracted centerline points by linear interpolation while the centerline of the vessel is a curve and it requires some post processing to smooth the tracing result. Figure 2.8 illustrates the “moving flat box” method.

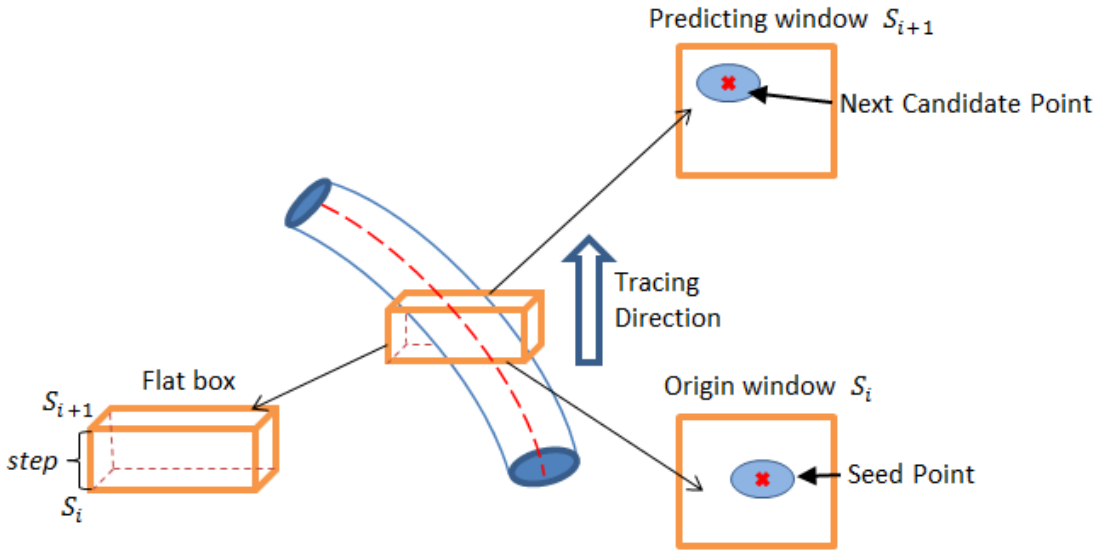


Figure 2.8: "Moving flat box" for 3D tracing. Surface Si contains the seed point which is the centroid of the cross section obtained in the previous step. The predicting window Si+1 is along the tracing direction. The centroid of the cross section of the vessel in the predicting window is considered as the next candidate point in the centerline [13].

Dileepkumar et al. [14] used breadth-first search to fully explore a specific branch. In this approach, the user provided a seed point to the algorithm to start the exploration. Then, boundaries of the cross section of the vessel in each successive slice was identified. After identifying boundaries, centroid and area of cross section of each region was identified. The centroid was used for tracing the vessel. The network was explored using breadth-first search with branch points as nodes and vessels as edges. This method has a limitation that, the exploration can proceed only in either direction of z-axis. That means, the algorithm cannot handle the case where the orientation of vessel is perpendicular to z-axis.

Lee et al. [8] presented an efficient 3-D thinning algorithm that iteratively removed voxels from the surface of the volume while preserving the number of connected components, cavities and holes in the original object. Kerschnitzki et al. [7] used Lee et al.'s algorithm to analyze and topologically quantify the osteocyte network in bone sections.

Palagyi et al. [15] presented a fully parallel surface thinning algorithm that could be used to extract the centerline of vasculature in medical images. In this method, each iteration was divided into six sub-iterations based on six main orthogonal directions in 3D and deletable points were removed iteratively until one-voxel wide centerlines were generated. Deletable points were given by a set of matching templates and were removed simultaneously in an iteration step. Lohou et al. [16] presented a six sub-iteration curve thinning algorithm based on P-simple points. This method deleted at least all the points deleted by Palagyi's method, while preserving the same end points.

Siddiqi et al. [17] introduced a new algorithm for simulating the eikonal equation, which offered significant computational advantages over earlier methods, and used it for shock tracking. Next they presented an efficient method for shock detection using eikonal equation and applied this method for skeleton generation. Cornea et al. [18] introduced the concept of hierarchal curve-skeleton and presented a method which computed an increasingly detailed curve-skeletons. This algorithm computed a repulsive force field over a discretization of the 3D object and used topological properties of the resulting vector field, to extract the curve skeleton.

3. METHODS

In this chapter, the methodology of this thesis is presented. To automatically trace and reconstruct the vascular structure in the dataset, the following steps were taken: generation of 3D volume from the 2D image sequence, image preprocessing, skeleton generation from the volume, dilation of the generated skeleton to reconstruct the original volume and quantitative analysis of the results. Each step is discussed in detail in the following sections.

3.1 Generation of 3D volume from 2D image sequence

The India ink dataset used in this research was obtained from the KESM brain atlas [1]. The raw 2 dimensional (2D) images were obtained from the coronal slices of a mouse brain. These 2D images were stacked together to obtain a raw 3 dimensional (3D) volume. The KESMStackProcessor, which was already available in BNL, was used for this conversion. The same area of each 2D slice was cropped and then the cropped slices were stacked together to generate the volume. The generated volume can be viewed in Paraview. Figure 3.1 shows an example of a 3D volume visualization in Paraview.

3.2 Image preprocessing

The next step is the preprocessing of the images. It consists of image filtering followed by thresholding of the filtered images.

The images obtained from KESM may contain some noise. To remove the noise from the images, the Wiener filter was used. Wiener filtering is used to reconstruct an

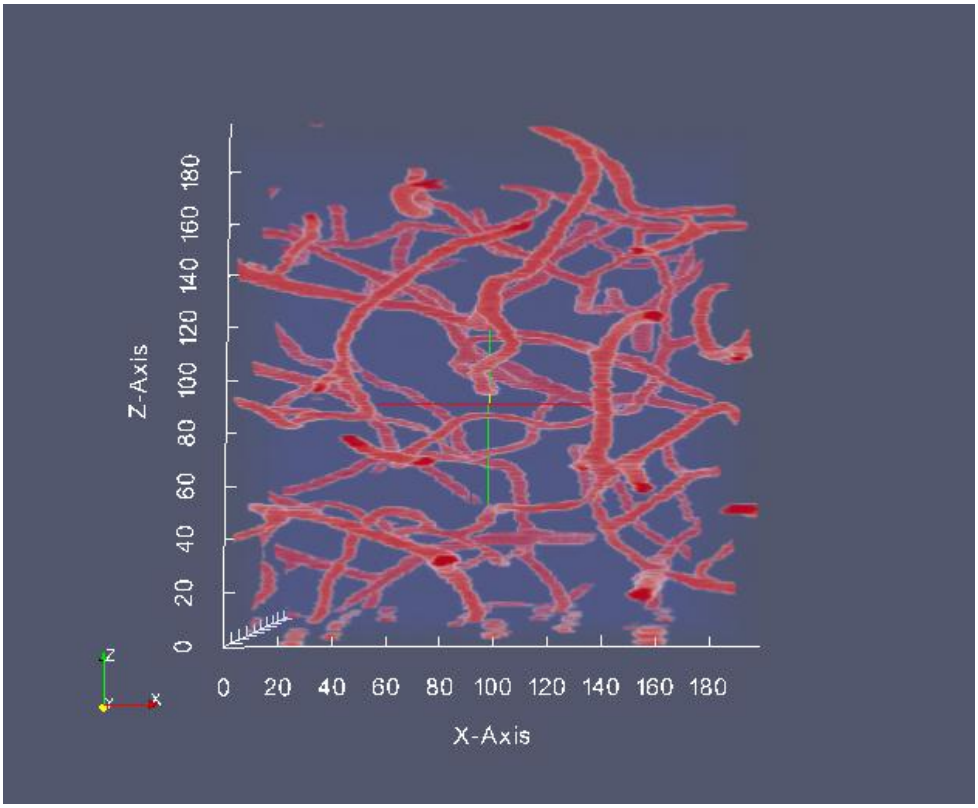
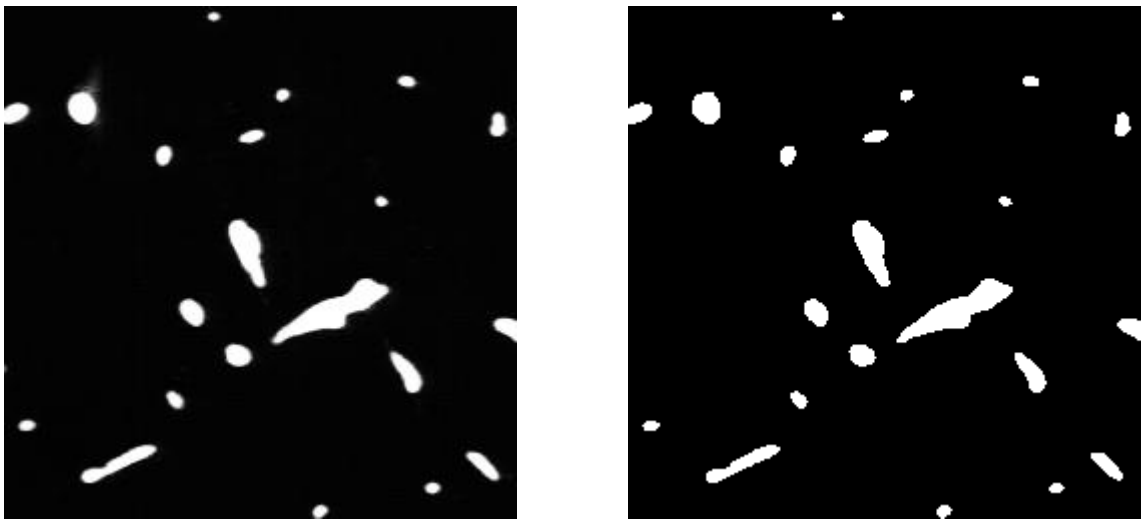


Figure 3.1: Visualization of a 3D data volume in Paraview

image that has been degraded by a degradation function and additive noise [19]. The Wiener filter minimizes the mean square error between the original image and the degraded image. MATLAB provides an implementation that uses a pixelwise adaptive Wiener method based on statistics obtained from local neighborhood of each pixel. This method estimates the local mean and variance around each pixel and the creates a pixelwise Wiener filter [5]. Noise variance is estimated as the average of all the local estimated variances.

After filtering, binary thresholding was applied on the images so that the foreground and background pixels can be easily separated. For this, Otsu's method [6]

was used. Ostu's method is a very efficient algorithm to binarize images that contain two classes of pixels. These images contain bi-modal histogram and the algorithm calculates the optimum threshold so that the intra-class variance in minimized. The algorithm exhaustively searches for the threshold where the intra-class variance is minimum. An implementation of this algorithm is provided in MATLAB. Figure 3.2 shows an original cross sectional image and image obtained after filtering and thresholding.



(a) (b)

Figure 3.2: Image preprocessing.(a) Original grayscale cross sectional image. (b) Image obtained after filtering and thresholding

3.3 Skeletonization

Skeletonization is the process used to reducing the foreground pixels in an image to a skeleton such that the connectivity and topology in the original image is preserved. A skeleton is a thin representation of the shape of an object in an image that ideally

passes through the middle of the object. Figure 3.3 shows an illustration of skeletonization in a 2 dimensional image.

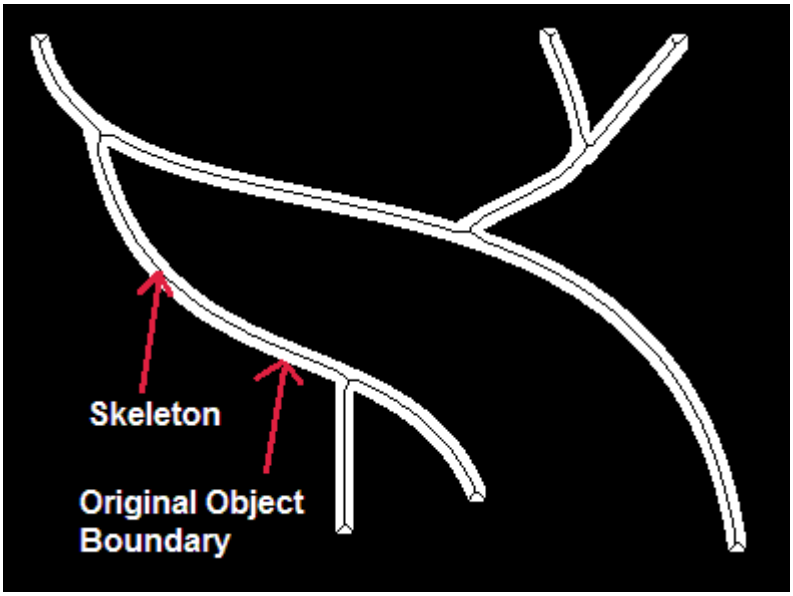


Figure 3.3: Illustration of skeletonization of an object. The black pixel line in the middle of white object represents the skeleton. Each point in the skeleton is equidistant to its boundaries. The skeleton preserves the topological properties of the object.

In 3-D space, skeleton is defined as the locus of the centers of all inscribed maximal spheres where these spheres touch the object's boundary at more than one point [8]. Figure 3.4 shows an illustration for the skeleton of a rectangular shape. Lee et al. [8] have discussed an efficient 3-D thinning algorithm that iteratively removes voxels from the surface of the volume while preserving the number of connected components, cavities and holes in the original object. Not all the boundary voxels can be removed without changing the topology. Figure 3.5 gives an illustration of a case in which

deletion of a boundary point breaks the object and thus, changes the number of connected components.

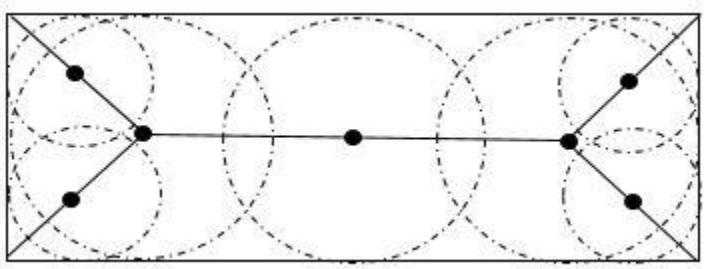


Figure 3.4: Skeleton of a rectangle. Skeleton is the locus of the centers of all inscribed maximal spheres where the spheres touch the boundary at more than one point. Adapted from [20].

Lee et al. [8] have proposed the following algorithm for thinning in 3-D space. First, they divide the thinning iteration into 6 subcycles according to 6 different types of border points. The different types of border points are N (north), S (south), W (west), E (east), U (up), B (bottom). The border point is of a particular type if the adjacent voxel in the corresponding direction is 0. For example, in Figure 3.6, if v is a border point, it is of type N, S, W, E, U or B if cubic index 13, 12, 10, 15, 4 or 21 is 0 respectively. In each subcycle, border points of only that particular type are considered for deletion.

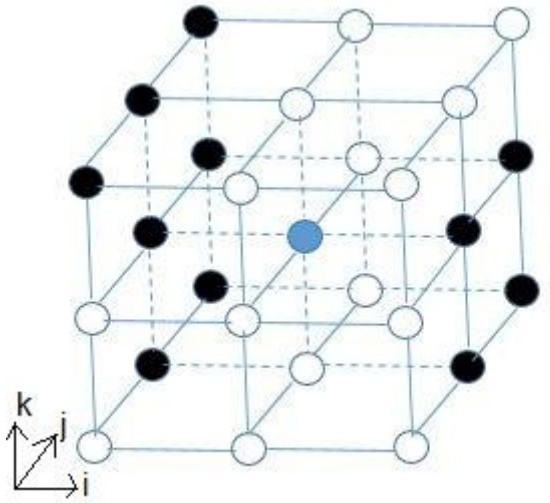


Figure 3.5: Illustration of a non-simple boundary voxel. All the colored circles depict voxels belonging to object. The circle marked in blue is a boundary voxel under consideration. Removal of this voxel breaks the connectivity of the object.

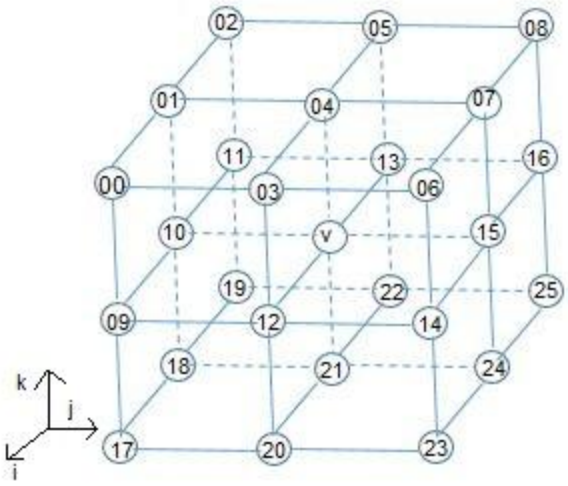


Figure 3.6: Indices of the 26-neighborhood of point v. Adapted from [8]

After obtaining the list of candidate points for removal, they are further classified as “simple” border points and “not-simple” border points. A border point is simple [8] if and only if satisfies the following conditions:

(1) Removal of v doesn’t change the Euler characteristic in the 3 x 3 x 3 neighborhood of v (*N(v)*) and

(2) The number of connected objects do not change in *N(v)* after removal or

(3) The number of holes do not change in *N(v)* after removal.

A hole in 3-D is different from a cavity in that it is not completely surrounded by foreground pixels. To check if a boundary voxel is simple or not, it is necessary to verify condition (1) and either of conditions (2) or (3). Only simple border points are considered for removal.

The Euler characteristic is defined as:

$$\chi(S) = O(S) - H(S) + C(S), \qquad (3.1)$$

where S denotes the set of voxels with value 1, O(S) denote the number of connected objects, H(S) is the number of holes and C(S) is the number of cavities of S. $\chi(S)$ is the Euler characteristic. To verify the condition (1), Lee et al. [8] have suggested a table look-up method. In this method, the Euler characteristic is considered only in the 3 x 3 x 3 local neighborhood of v and in this case, the Euler characteristic is denoted by G(S). This neighborhood *N(v)* is divided into eight overlapping octants of size 2 x 2 x 2. Each octant contains 8 voxels. So, total number of different combinations in each octant can be $2^8 = 256$, which can be stored in a table. The Euler characteristic for v is the sum of the Euler characteristics of the eight octants. In this way, change in the Euler

characteristic after removal of v can be calculated and if this change is 0, then we verify the next condition. Otherwise, the point is removed from the list of candidate points.

To verify the second condition, Lee et al. [8] have determined change in the number of connected components in *N(v)* after removing v. This can be done efficiently by using an octree data structure and recursive depth first search algorithm. If there is no change in number of components in *N(v)*, then v is a simple boundary point. Otherwise, v is removed from the list of candidate points.

Finally, all the voxels with only exactly one neighbor are removed from the list of candidate points. These voxels are end-points and should not be removed from the skeleton.

The list of points obtained after the above procedure can be removed from the volume. But, because the procedure is carried out in parallel, there can still be some points [8] that change the connectivity in the neighborhood. For example, consider Figure 3.7. All white points in 3.7a and 3.7b can be deleted as they are simple border points of U-type. However, simultaneous removal of these points will cause 3.7a to be broken into two components and 3.7b to be completely removed. Therefore, it is necessary perform re-checking for the points to be removed. Let R denote the set of points obtained above and Q is the set of points obtained after removing R from the points in the skeleton S obtained so far. Then, for each point v in R, it is verified that removal of v doesn't change the connectivity in Q. If the connectivity in Q is not changed then v is removed from S.

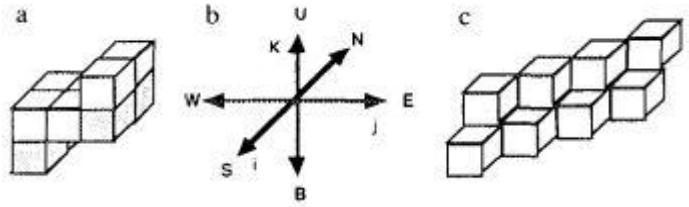


Figure 3.7: Illustration of the necessity to perform rechecking. White points show the simple border points of type U. (a) Object which gets broken on removal of all the white points (b) Types of border point. This example illustrates type U. (c) Object which gets completely eliminated on removal of white points [8][1].

Kerschnitzki et al.[7] have provided an efficient implementation for the above algorithm in MATLAB. They have used this algorithm to analyze and topologically quantify the osteocyte network in bone sections.

After obtaining the skeleton, we generate a VTK trace from the skeleton. This file contains a list of points present in the skeleton and information about their connectivity in their 3 x 3 x 3 local neighborhood. The list contains coordinate information of the points. To obtain connectivity information, we look into the 26 neighbors of a point v belonging to the skeleton and connect v to all the neighbors which (1) belong to the skeleton and (2) have not already been connected to v in the VTK file. We processed the list of skeleton points in increasing order of their index and connect the neighbor point only if the index of neighbor is greater than the index of the point. This VTK file can be viewed in Paraview, an image visualization software. We can also overlay the original volume and the VTK file in Paraview for visual inspection. Figure

[1] Reprinted from CVGIP: Graphical Models and Image Processing, vol. 56, T.C. Lee,R.L. Kashyap,C.N. Chu, Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms, pp. 462-478., Copyright (1994), with permission from Elsevier.

3.8 shows an illustration of overlay of original volume and the skeleton. The red region in the image represents the vasculature structure in the original volume. The white lines represent the skeleton.

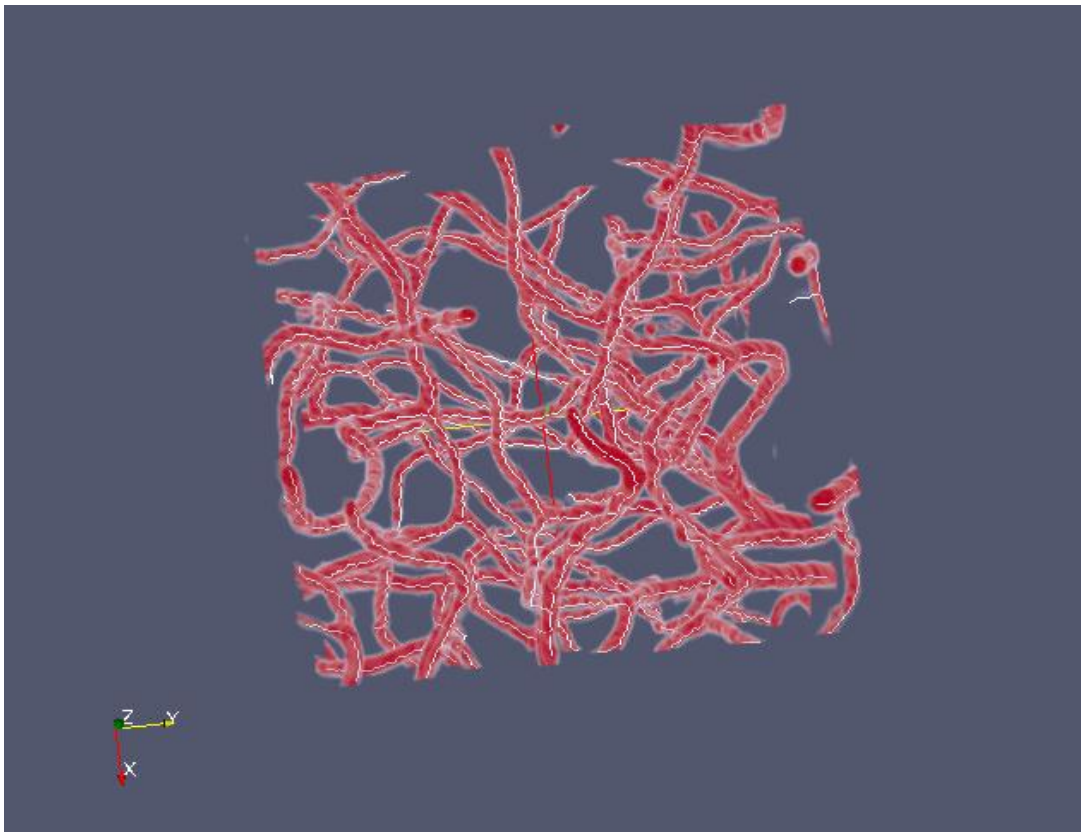


Figure 3.8: Overlay of original volume and its skeleton using Paraview. The original volume is represented by red color and the skeleton by white lines passing through the middle of the red volume.

3.4 Reconstruction of the original volume

To validate the results, we reconstruct the volume from the generated skeleton and compare with the original volume to calculate the accuracy. To reconstruct the

volume, we dilated the skeleton with an appropriate radius using a sphere as structuring element. The equation for the structuring element is:

$$\sqrt{x^2 + y^2 + z^2} \leq r, \tag{3.2}$$

where r is the radius of the sphere.

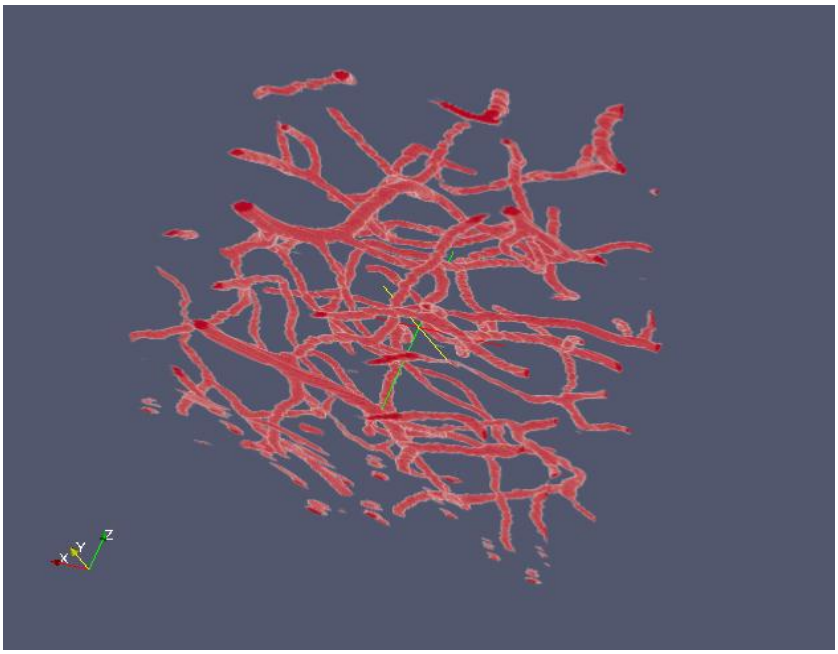
But, we cannot perform dilation with a constant radius throughout the volume because different vessels have different diameter and even the same vessel may have different diameters in different regions. To solve this problem, we estimate the vessel diameter at each point in the skeleton using the original volume. For diameter estimation, we calculate the diameter in the x-direction, y-direction and z-direction in the original volume for each point v in the skeleton. We get the minimum of the 3 diameters and then calculate the radius r at that point. Figure 3.9 illustrates an example of original volume and the volume reconstructed from skeleton.

For quantitative analysis of the results, we will calculate the following: accuracy, precision, recall and F-measure.

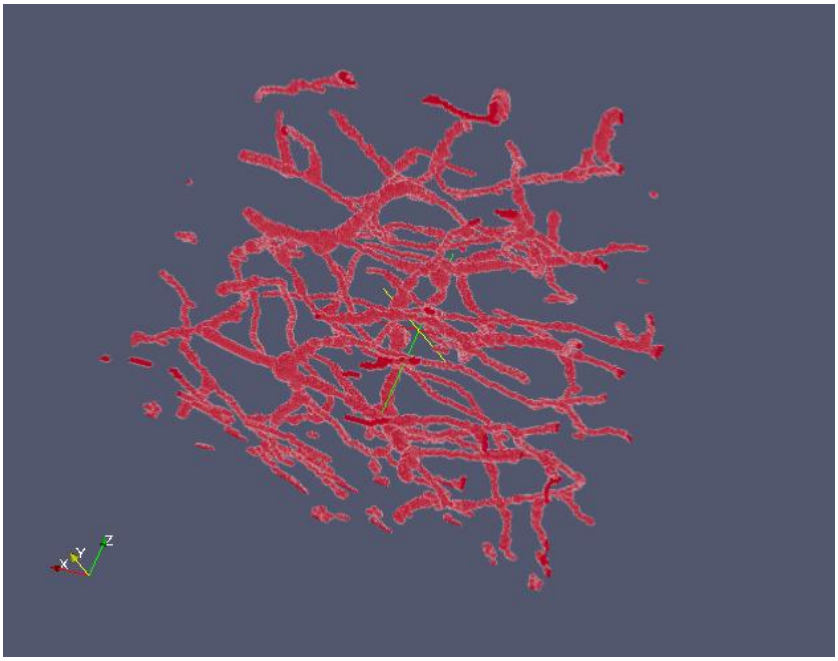
$$Accuracy = \frac{number\ of\ voxels\ that\ match\ in\ both\ volume}{total\ number\ of\ voxels\ in\ volume} \tag{3.3}$$

The numerator is the sum of all the voxels such that if the voxel belong to the object in reconstructed volume, the corresponding object in the original volume also belongs to the object and if the voxel does not belong to the object in the reconstructed volume, the corresponding voxel does not belong to the object in the original volume as well.

$$Precision = \frac{Number\ of\ voxels\ that\ belong\ to\ the\ object\ in\ both\ the\ volumes}{Number\ of\ voxels\ that\ belong\ to\ object\ in\ reconstructed\ volume} \tag{3.4}$$



(a) Original volume



(b) Reconstructed volume

Figure 3.9: Visual comparison of original and reconstructed volume

Precision gives the ratio of the actual number of voxels belonging to object retrieved in the reconstructed volume to the number of voxels belonging to object present in the reconstructed volume.

$$Recall = \frac{Number\ of\ voxels\ that\ belong\ to\ the\ object\ in\ both\ volume}{Number\ of\ voxels\ that\ belong\ to\ the\ object\ in\ original\ volume} \quad (3.5)$$

Recall gives the ratio of the actual number of voxels belonging to object retrieved in the reconstructed volume to the number of voxels belonging to object present in the original volume.

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (3.6)$$

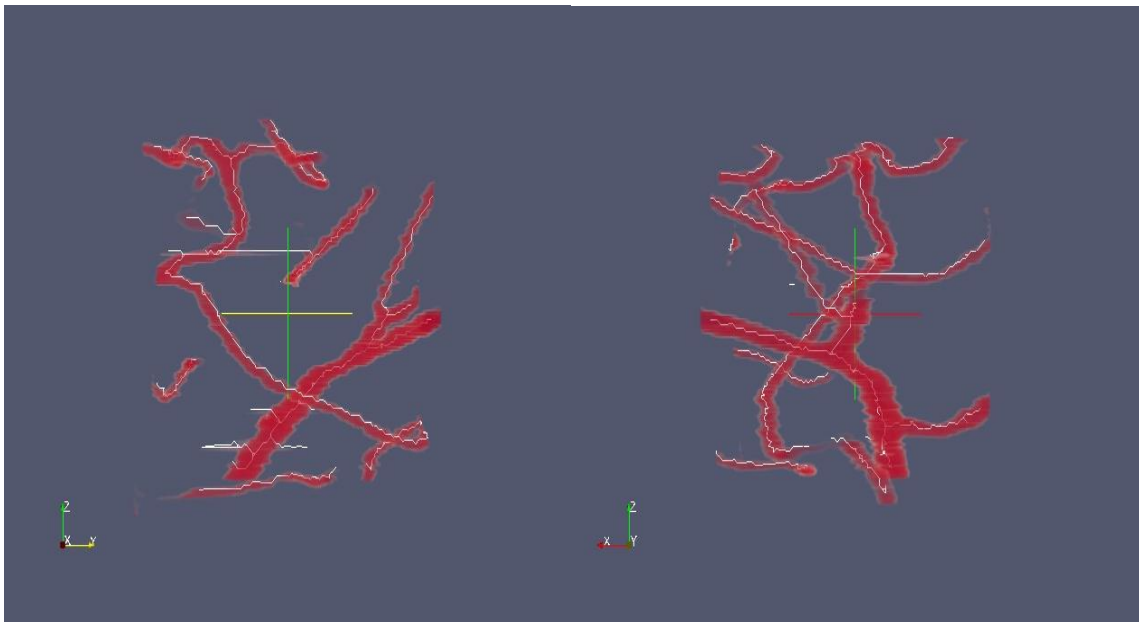
F-measure is the harmonic mean of Precision and Recall.

4. RESULTS AND ANALYSIS

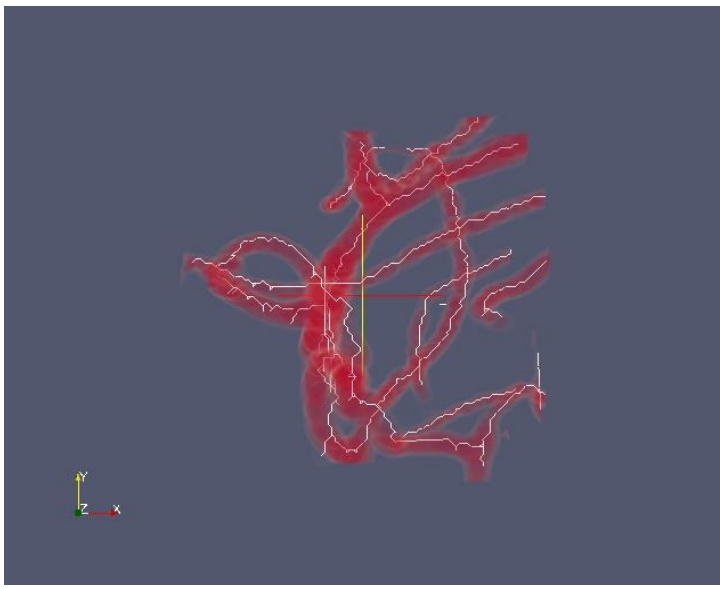
In the previous chapter, the method for generating the skeleton and reconstructing the volume was presented. In this chapter, results and quantitative analysis of this approach is presented. The analysis was done on sub-networks of the KESM India Ink data set. First, the results of the skeleton generated from the volume are presented. Next, the results of reconstructed volume are presented. Finally, the accuracy, precision, recall and F-measure of the reconstructed volume is evaluated and the running time of the algorithm is analyzed.

4.1 Results of skeleton generation

The analysis was done on sample sub-networks of size 100×100×100, 128×128×128, 200×200×200, 256×256×256 and 512×512×512. First, the skeleton was generated using the algorithm described in the previous section, then the skeleton was used to generate a VTK trace. Finally, the original volume and the VTK trace were overlaid in paraview for visual inspection of the skeleton. Figures 4.1 to 4.7 show the 3-dimensional view of the tracing results of volumes of different sizes. In these figures, the white curves within the red vessels represent the VTK trace. Visual inspection shows that the vessel in the datasets were fully explored and the skeleton represents the medial axis of the vessels.

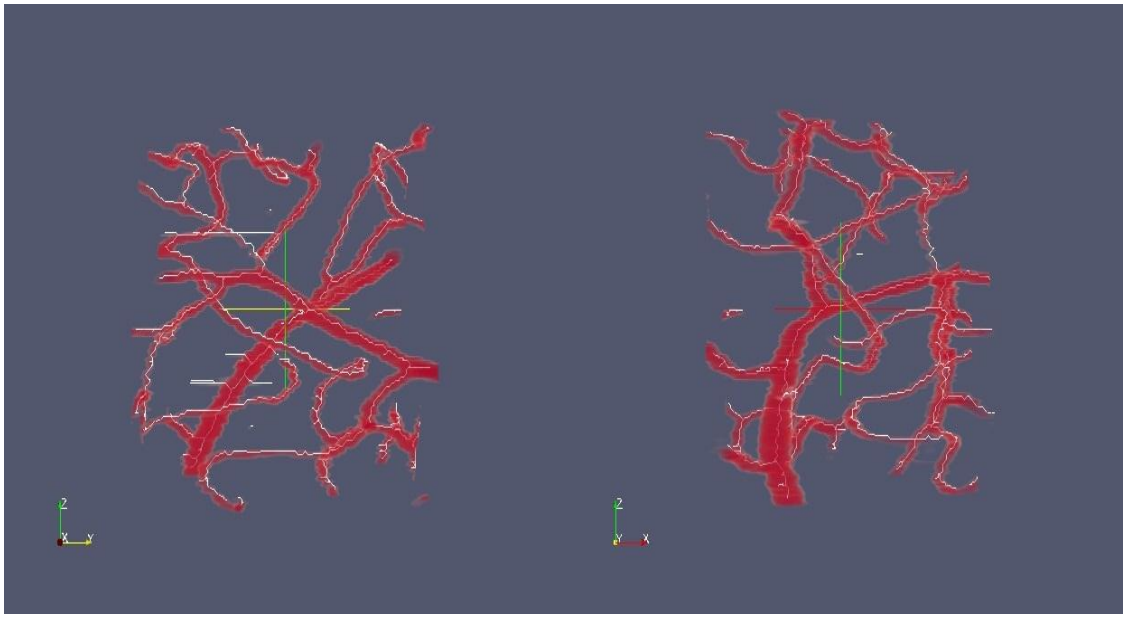


(a) (b)

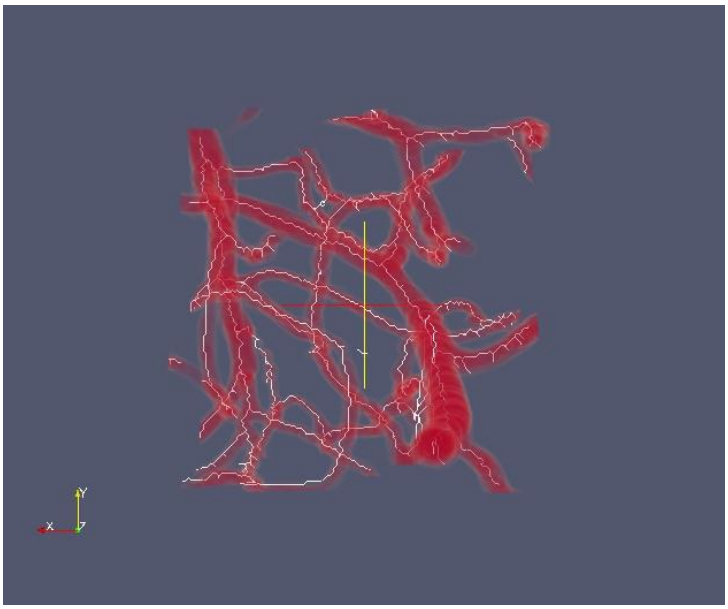


(c)

Figure 4.1: The vascular tracing results of KESM 100×100×100 volume. (a) View along the x-axis. (b) View along the y-axis. (c) View along the z-axis

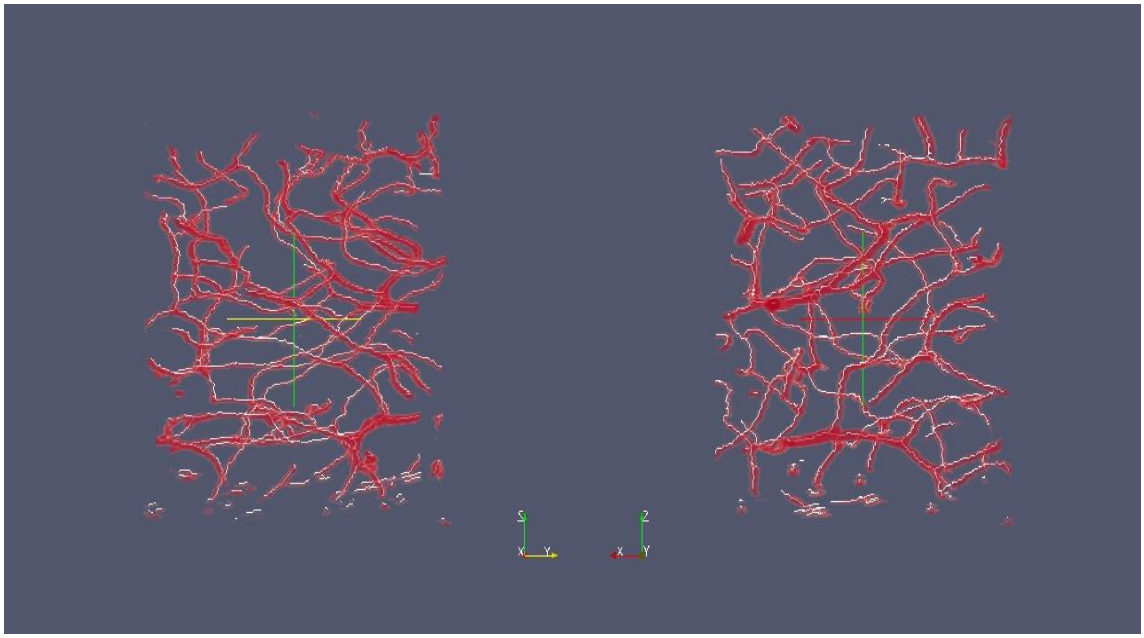


(a) (b)

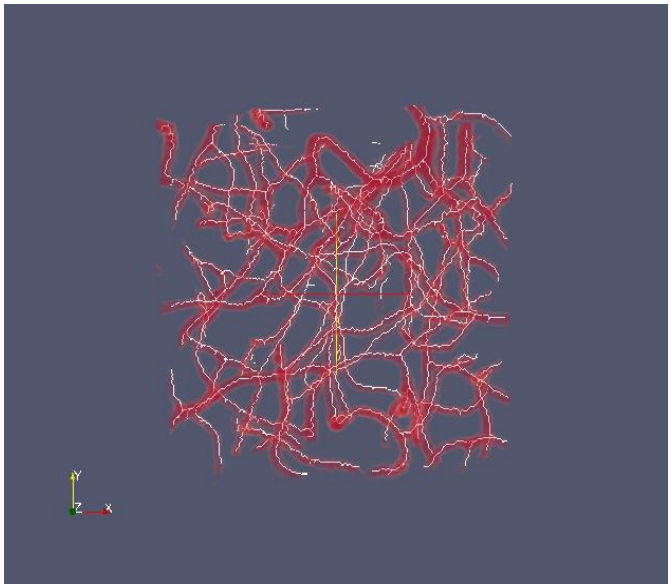


(c)

Figure 4.2: The vascular tracing results of KESM 128×128×128 volume. (a) View along the x-axis. (b) View along the y-axis. (c) View along the z-axis

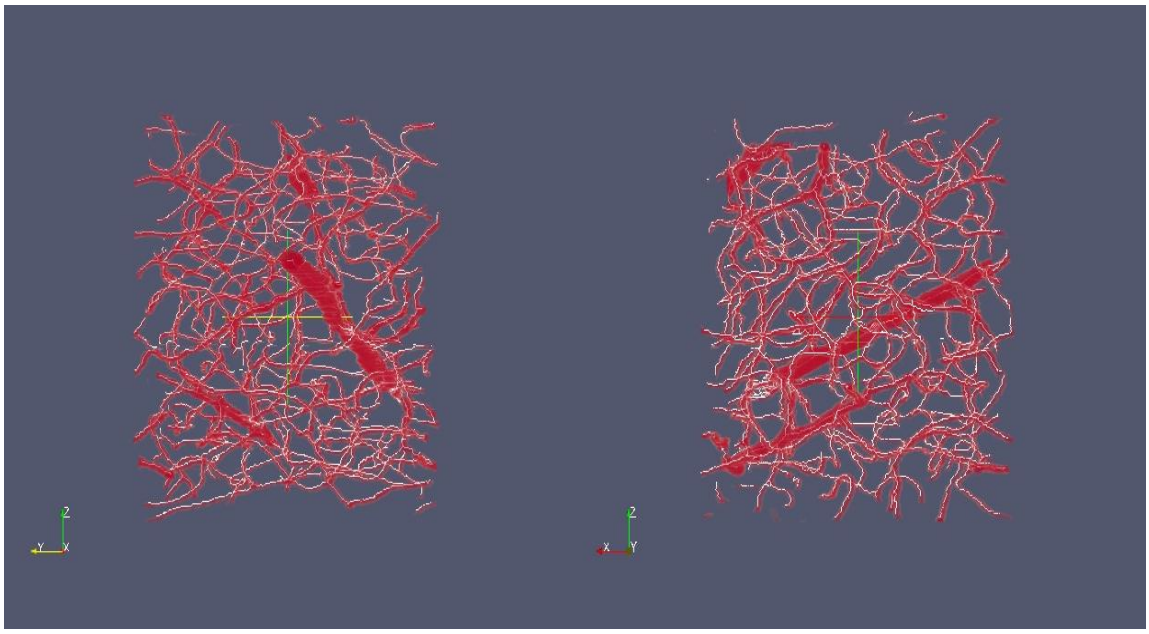


(a) (b)

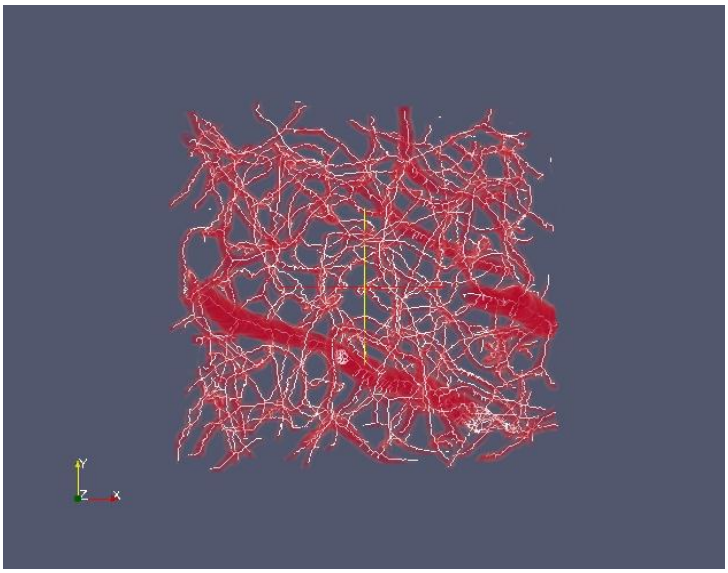


(c)

Figure 4.3: The vascular tracing results of KESM 200×200×200 volume. (a) View along the x-axis. (b) View along the y-axis. (c) View along the z-axis



(a) (b)



(c)

Figure 4.4: The vascular tracing results of KESM 256×256×256 volume. (a) View along the x-axis. (b) View along the y-axis. (c) View along the z-axis

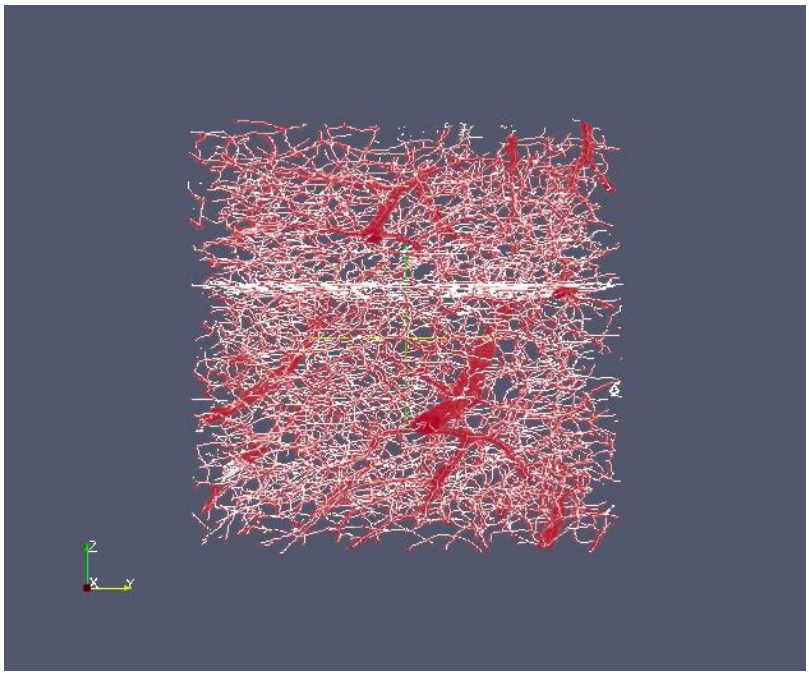


Figure 4.5: View along x-axis for tracing results of KESM 512×512×512 volume

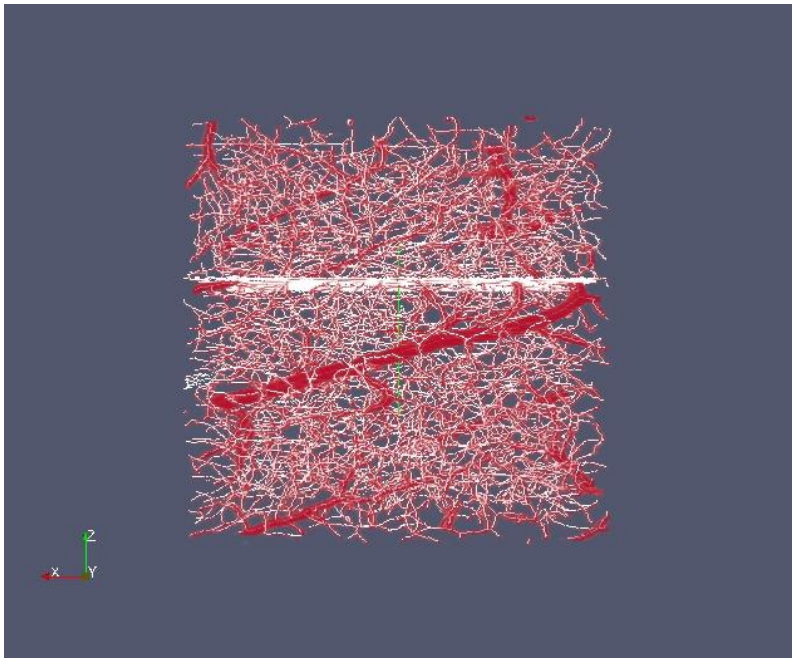


Figure 4.6: View along y-axis for tracing results of KESM 512×512×512 volume

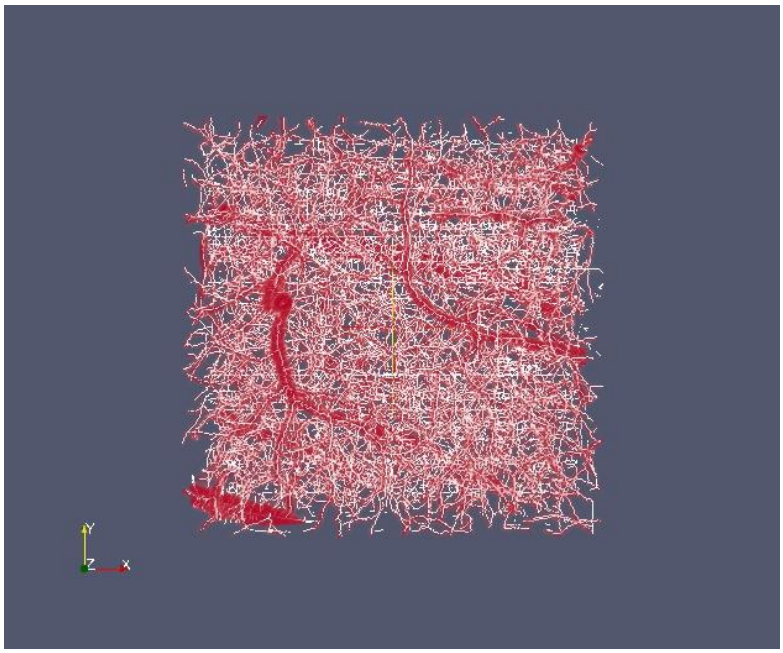


Figure 4.7: View along z-axis for tracing results of KESM 512×512×512 volume

From the skeleton, it is easy to collect geometrical statistics of the datasets. The size of the skeleton is defined as the number of points contained in the skeleton. A branch point (or a node) is defined as a point in the skeleton that is connected to more than two points in the 26 neighborhood space of that point. This is the point at which either a vessel bifurcates, two vessels merge together or a new branch emerges from an existing vessel. An end point in defined as a point which is connected to only one other point in the 26 neighborhood space. This is the point at which a vessel terminates. Mayerich et al. [21] considered any vessel that terminates without branching as an error because in microvascular networks, all segments must be connected in order to allow blood flow. We have ignored the end points at the volume boundaries and only counted

end points contained inside the volume. Table 4.1 shows the statistics collected during the skeleton generation.

Table 4.1: Statistics collected for volumes of different sizes.

Volume size	Skeleton size	Branch points	End points	Error (%)
100^3	1007	87	31	3.08
128^3	2200	238	70	3.18
200^3	6602	609	217	3.29
256^3	17889	2287	745	4.16
512^3	163191	21598	9463	5.80

Using these data, we can calculate vessel termination error for each volume as the ratio of the number of end points to the skeleton size. Mayerich et al. [21] have reported that these errors accounted for approximately 3.2% of the network. We have also achieved almost similar error. These errors can be due to artifacts either in the scanned images or in the generated skeleton. Figure 4.8 shows the plot of the skeleton size against the volume size on a logarithmic scale. The plot shows a linear relationship between the number of points obtained in the skeleton and the volume size. Figure 4.9 shows the plot of the number of branch points against the skeleton size on a logarithmic scale. This plot shows a sub-linear relation between the number of branch points in the skeleton and the size of the skeleton.

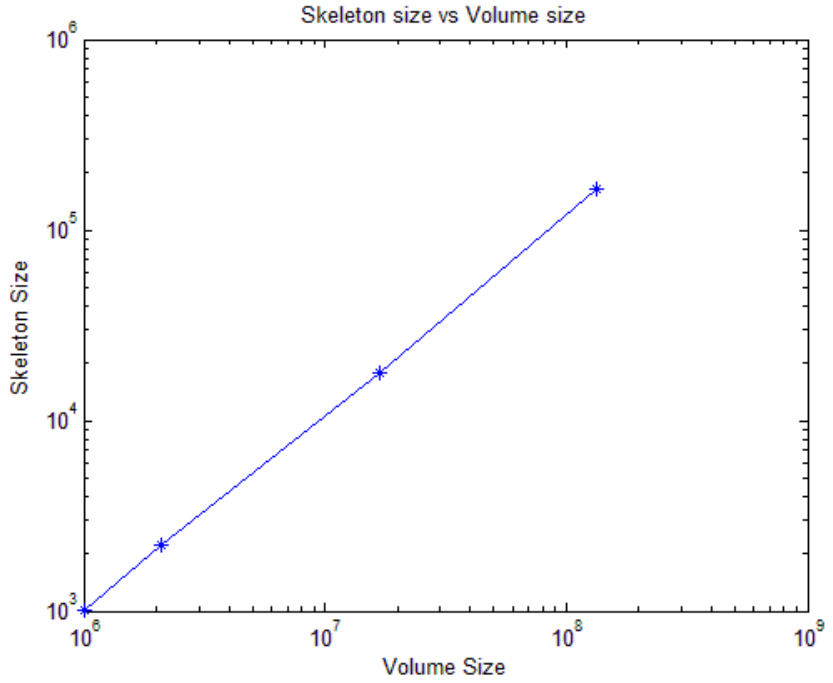


Figure 4.8: Skeleton size versus volume size

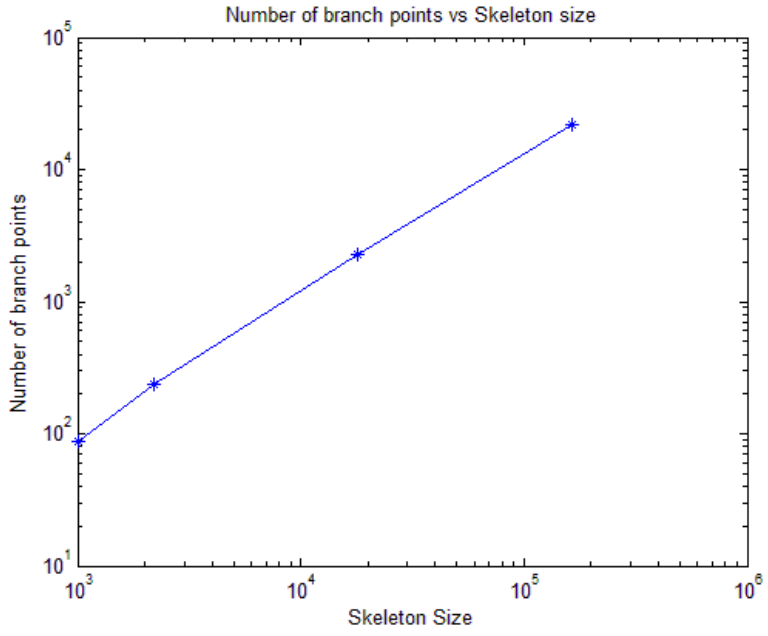
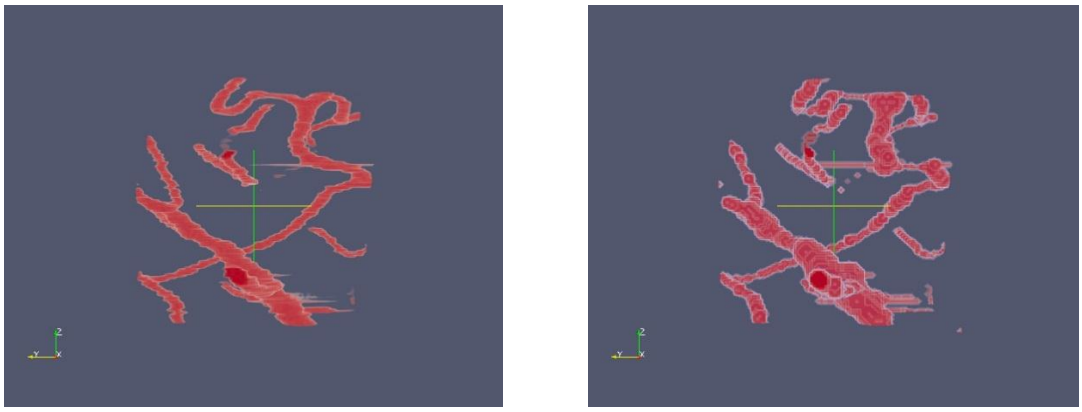


Figure 4.9: Number of branch points versus skeleton size

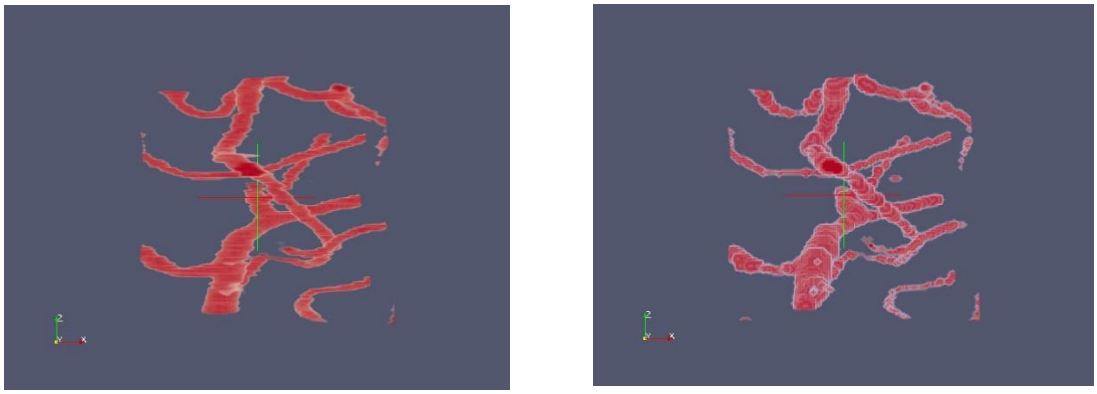
4.2 Results of volume reconstruction

For reconstruction of the vascular structure in the KESM dataset, the skeleton was dilated using the radius obtained by estimation as explained in section 3.4. For quantitative analysis of the results, the reconstructed volume was compared with the original volume. Accuracy, precision, recall and F-measure were calculated using the equations described in section 3.4. Figures 4.10 to 4.14 show the visual comparison of the original and reconstructed volumes of sizes 100×100×100, 128×128×128, 200×200×200, 256×256×256 and 512×512×512 respectively. Figure 4.10 shows the views along x, y and z axis. Figures 4.11 to 4.14 show general 3-dimensional views. From the visual inspection, it looks like the reconstructed volumes are very similar to the original volumes.

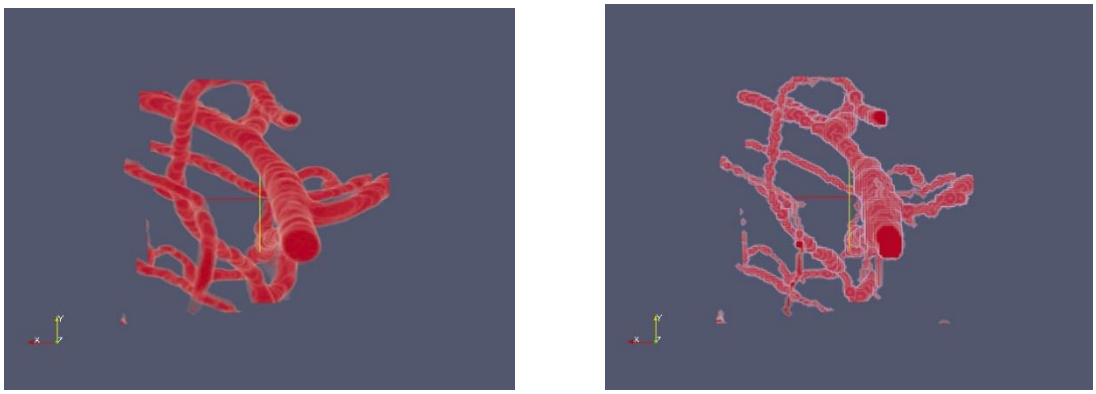
For evaluation and validation of the results, quantitative measurements at the voxel level were performed. To calculate accuracy, we first calculated the number of voxels that do not match in the original and reconstructed volume. The volumes were converted to binary volumes in which 1 represented voxel belonging to foreground (object) and 0 represented voxel belonging to background. By applying exclusive-or operator between the original and reconstructed volume, we found the error in the reconstructed volume. From this, we calculated the voxels that match in both the volumes and the accuracy was calculated by dividing the total number of voxels that match by the total number of voxels present in the volume. Precision was calculated by applying the AND operator between the two volumes to obtain the number of voxels belonging to the object in both the volumes and then dividing this by the number of the voxels belonging to object in



(a) View along x-axis

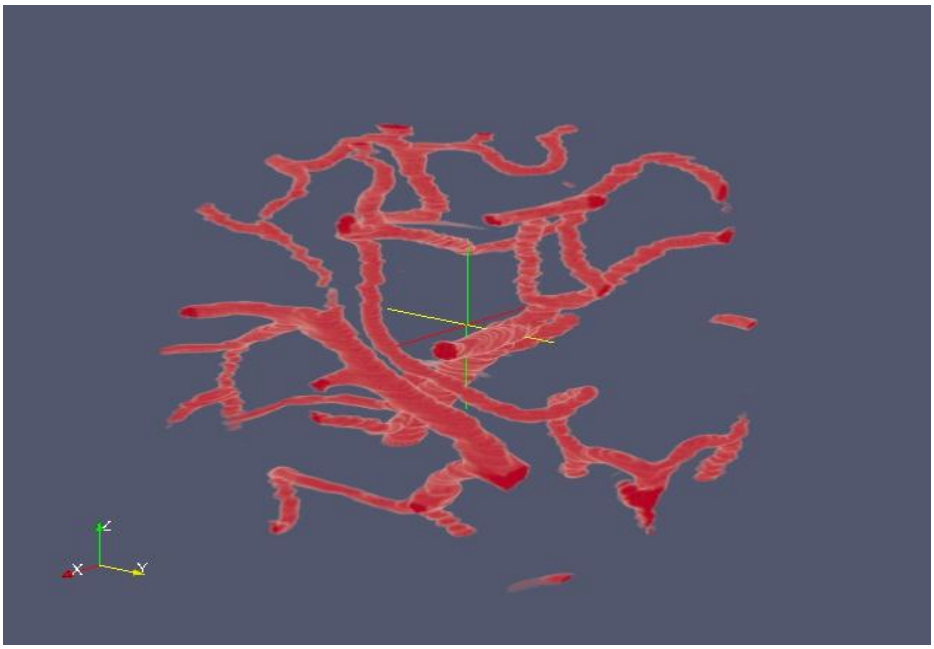


(b) View along y-axis

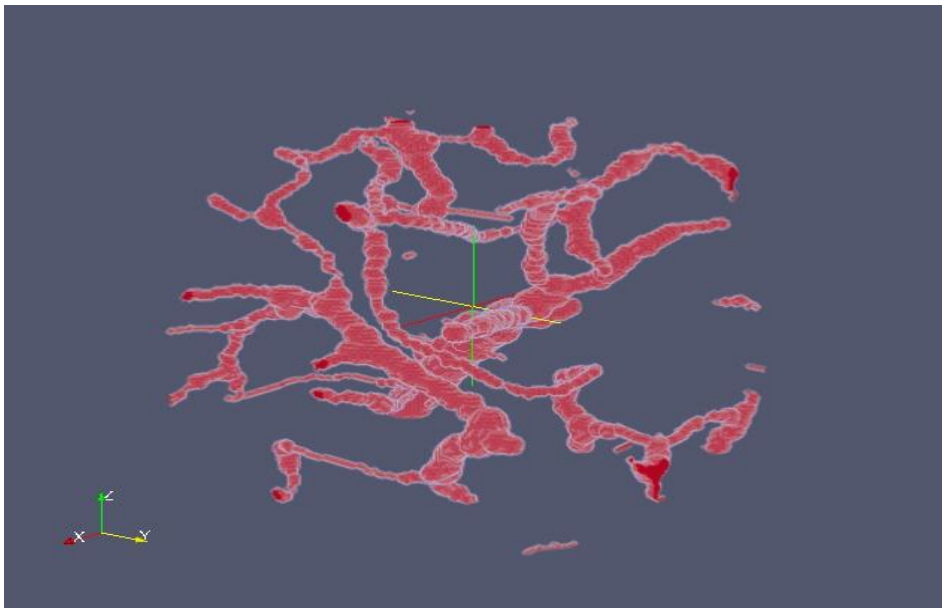


(c) View along z-axis

Figure 4.10: Original (left) and reconstructed (right) volumes of size 100×100×100

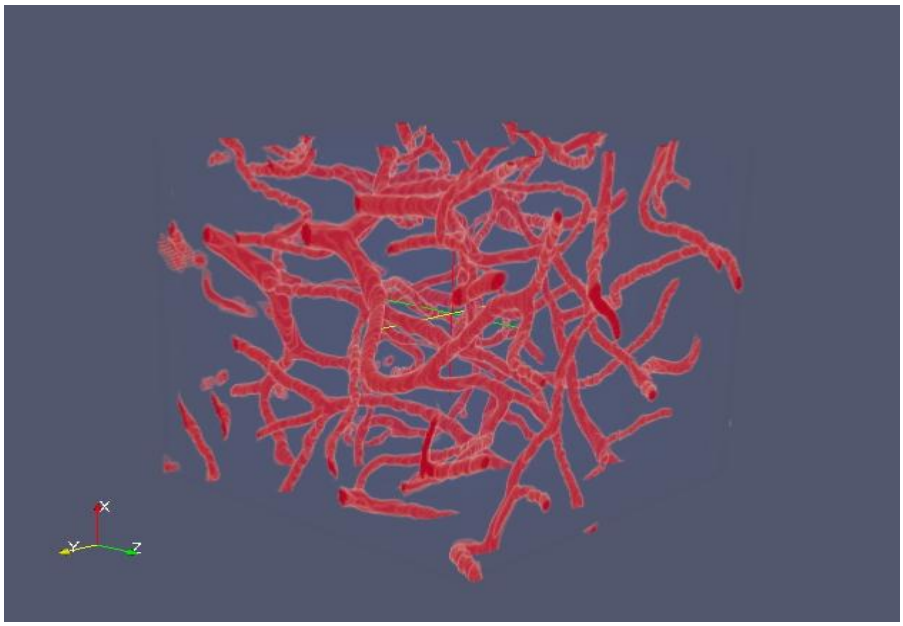


(a) Original Volume

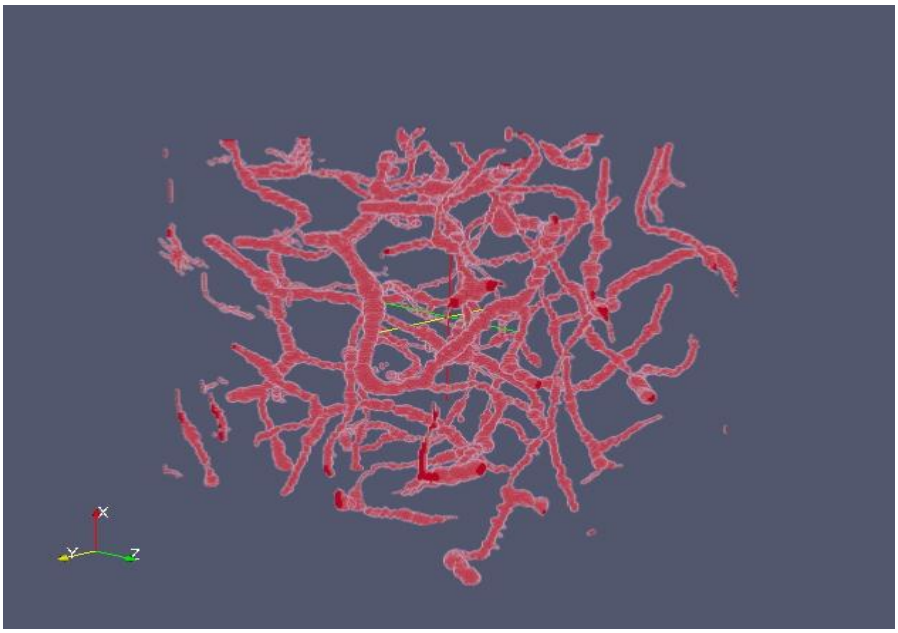


(b) Reconstructed Volume

Figure 4.11: Original and reconstructed volumes of size 128×128×128

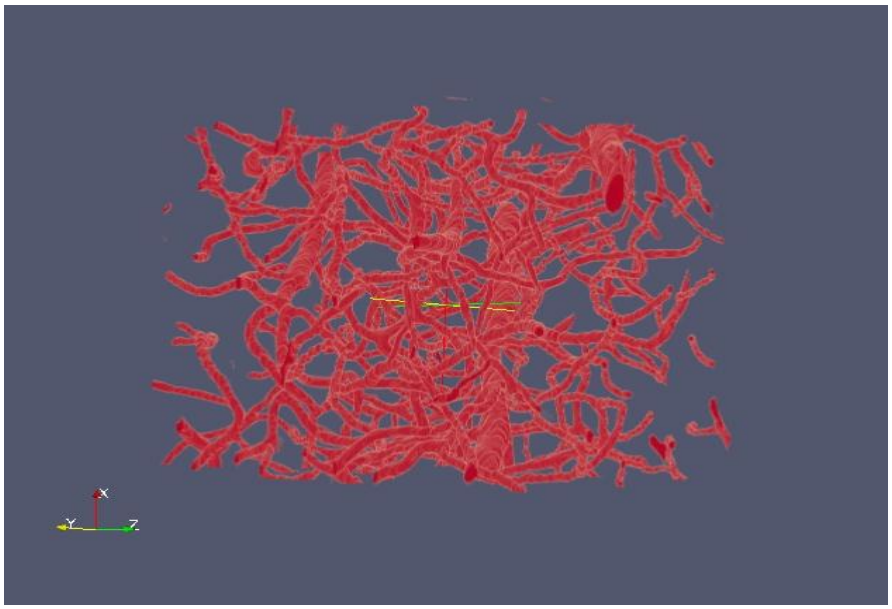


(a) Original Volume

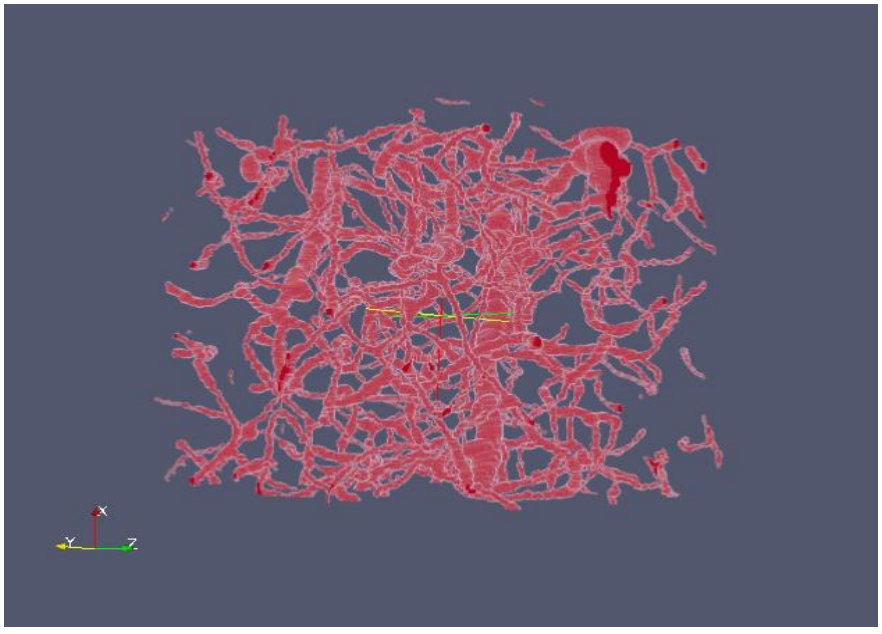


(b) Reconstructed Volume

Figure 4.12: Original and reconstructed volumes of size 200×200×200

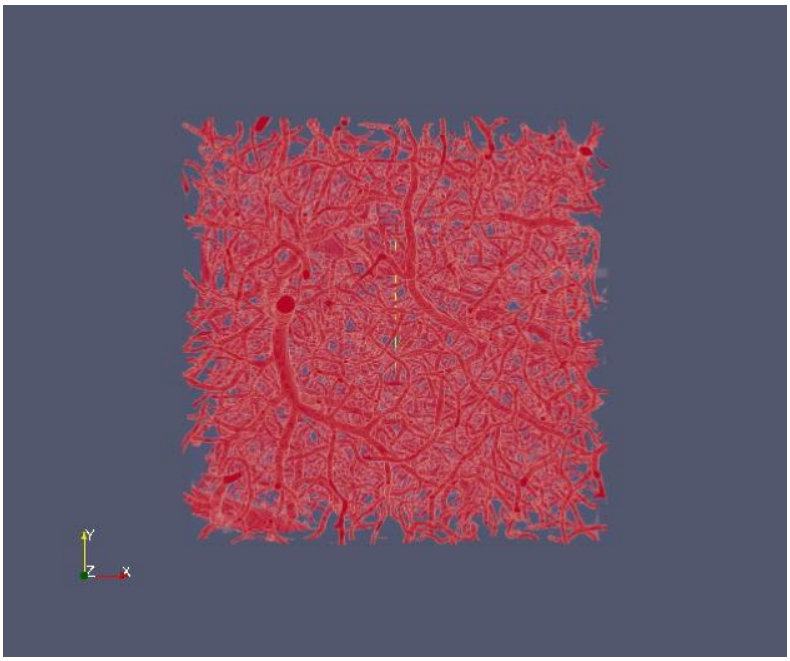


(a) Original Volume

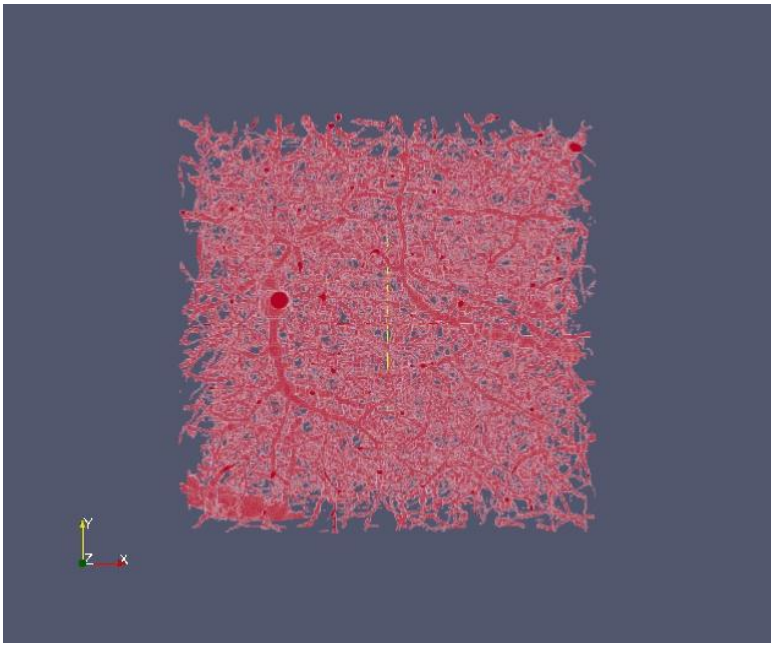


(b) Reconstructed Volume

Figure 4.13 Original and reconstructed volumes of size 256×256×256



(a) Original Volume



(b) Reconstructed Volumes

Figure 4.14 Original and reconstructed volumes of size 512×512×512

the reconstructed volume. Similarly, recall was obtained by diving the number of voxels belonging to the object in both the volumes by the number of voxels belonging to the object in the reconstructed volume. Table 4.2 shows the results for different volumes.

Table 4.2: Quantitative analysis of the reconstructed volumes

Volume size	Accuracy (%)	Precision (%)	Recall (%)	F-measure
100^3	98.49	77.47	72.99	75.16
128^3	98.43	75.58	76.67	76.12
200^3	98.74	73.92	64.10	68.66
256^3	98.40	80.98	62.46	70.52
512^3	98.27	80.97	62.50	70.55

Figure 4.15 shows the plot of the results shown in Table 4.2. From the plot, it is clear that the accuracy is very good (~98.5%) and the accuracy is not affected by increasing the size of the data volume. This shows that the reconstructed models are very similar to the original vascular network in the dataset and that our method can trace and reconstruct very large KESM datasets accurately and efficiently.

We also obtained very good precision using our method. Precision for dataset of size 100×100×100 voxels was 77.47 and the precision for dataset of size 512×512×512 voxels was 80.97. This shows that in the reconstructed models, most of the voxels belonging to foreground also belong to foreground in the corresponding KESM dataset.

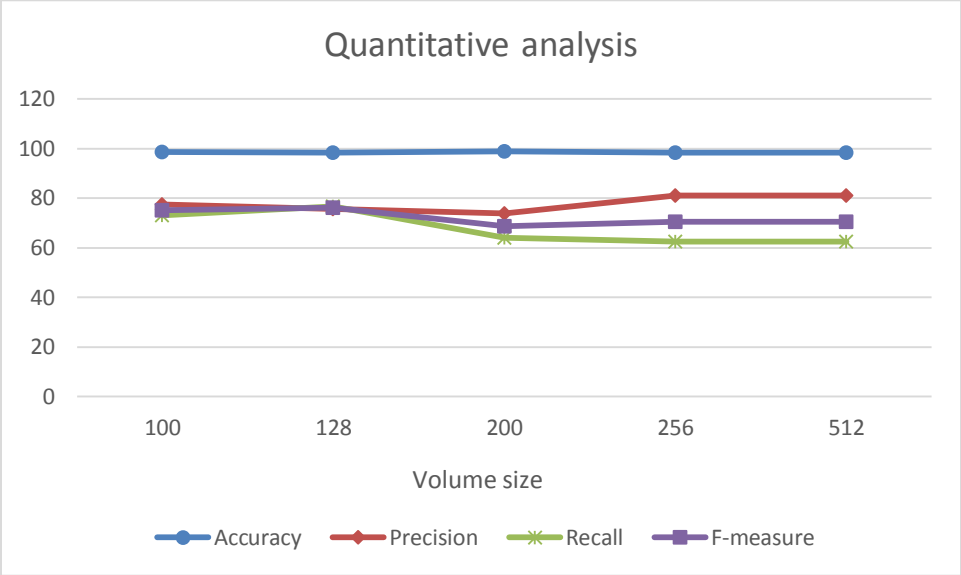


Figure 4.15: Quantitative analysis of the reconstructed volume. The horizontal axis represents the different volume sizes.

Precision is, in general, better than recall. This can be explained as the vessels in the datasets are not completely circular. Since we used the minimum radius obtained along the x-axis, y-axis and z-axis, we were not able to retrieve all the points present in the original dataset. Therefore, recall is slightly lower than precision.

F-measure, which measures the harmonic mean between precision and recall, doesn't vary much by increasing size of the volume. From these results, we can conclude that the skeletonization based tracing method is very accurate and can efficiently trace very large volumes.

Figures 4.16 to 4.20 show the histograms of diameter for volumes of different sizes. From these histograms, we can see that different vessels have different diameters.

We can also see that the diameter distribution of vessels is very similar for volumes of different sizes.

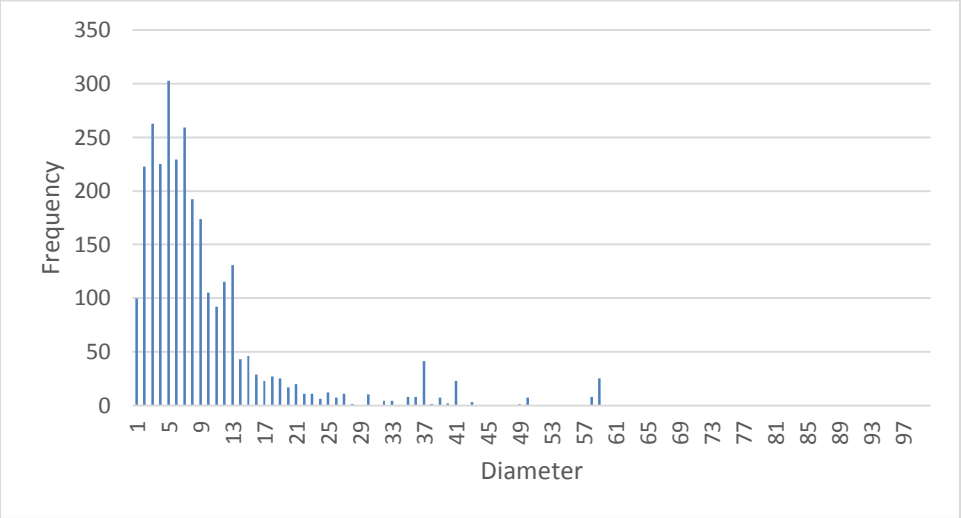


Figure 4.16: Histogram of diameter for volume of size 100×100×100

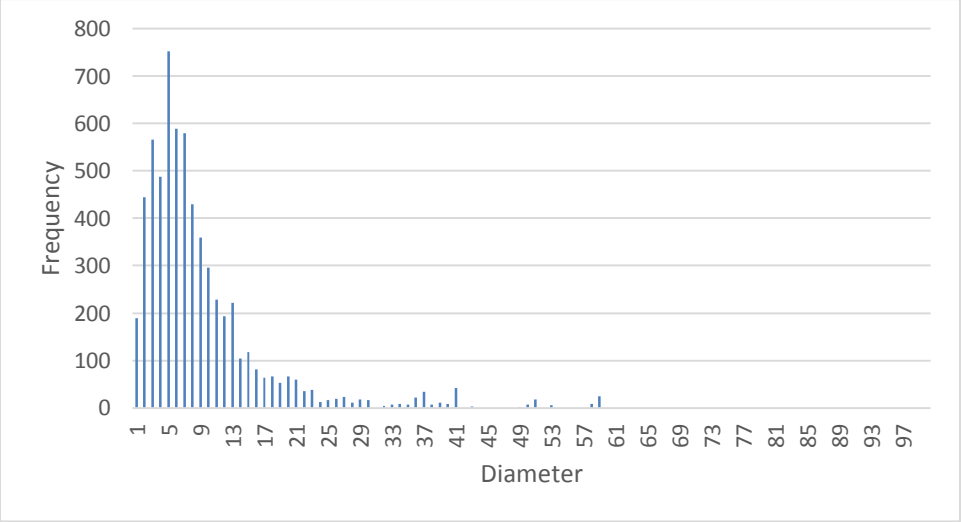


Figure 4.17: Histogram of diameter for volume of size 128×128×128

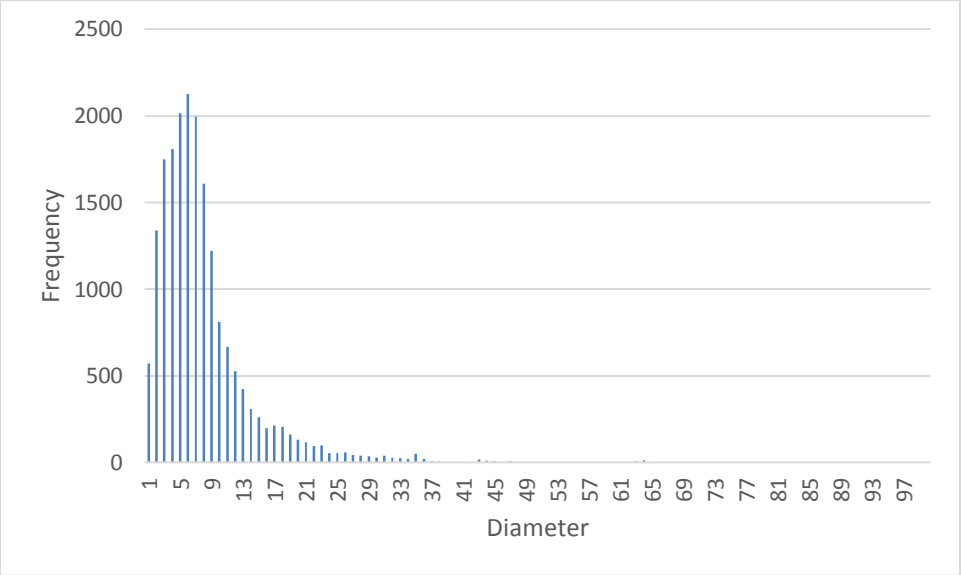


Figure 4.18: Histogram of diameter for volume of size 200×200×200

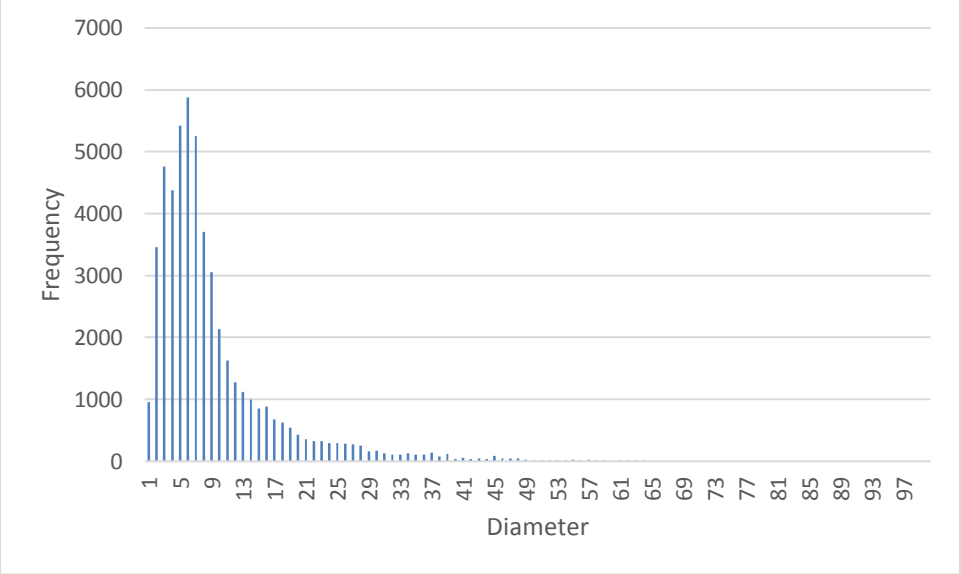


Figure 4.19: Histogram of diameter for volume of size 256×256×256

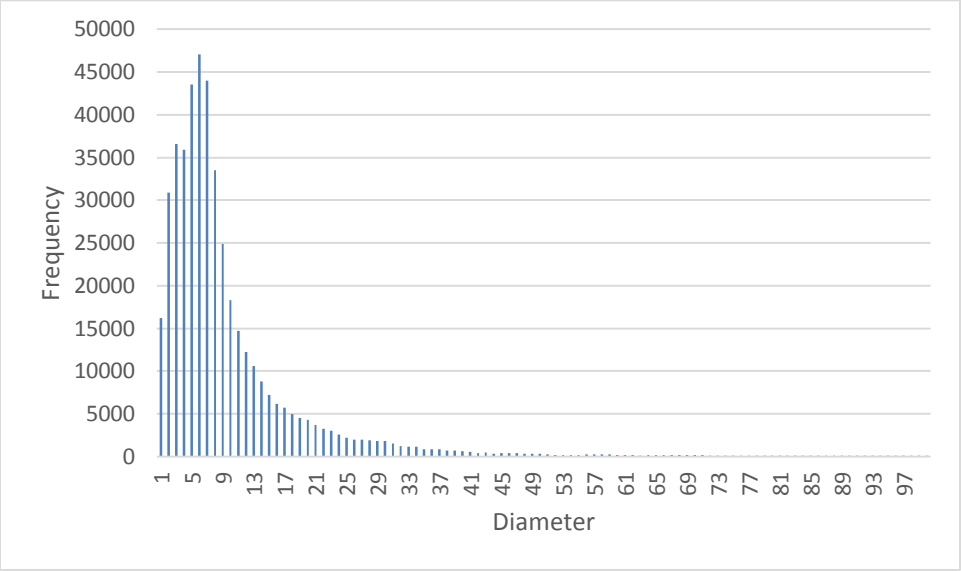


Figure 4.20: Histogram of diameter for volume of size 512×512×512

Table 4.3 shows the KL divergence between different histograms. KL divergence is a non-symmetric measure of the difference between two probability distributions. For example, KL divergence of the histogram of 512^3 volume from the histogram of 256^3 volume is 0.0817, whereas KL divergence of the histogram of 256^3 volume from the histogram of 512^3 volume is 0.0107. It can be seen from the table that the KL divergence between different histograms is very small which shows that the diameter distribution of vessels is very similar for different volumes.

Table 4.3: KL divergence between different histograms

	100^3	128^3	200^3	256^3	512^3
100^3	0	0.0306	0.0864	0.1196	0.0999
128^3	0.0131	0	0.0431	0.0601	0.0502
200^3	0.0550	0.0342	0	0.0261	0.0339
256^3	0.0531	0.0262	0.0219	0	0.0187
512^3	0.0292	0.0091	0.0088	0.0107	0

4.3 Analysis of tracing speed

In this section, we will analyze the running time of our method. Since our method will be applied on a very large dataset, the tracing speed has to be very good. The experiments were performed using single core on a PC with Intel Core 2 Duo (2.00 GHz) processor under Windows 7 operating system. The time calculated was the time to generate the skeleton from the raw datasets. The time taken on the five volumes mentioned in section 4.1 were 15.67, 20.42, 38.79, 104.45 and 1010.64 seconds and the total number of voxels in the skeletons generated in these experiments were 1007, 2200, 6602, 17889 and 163191 respectively.

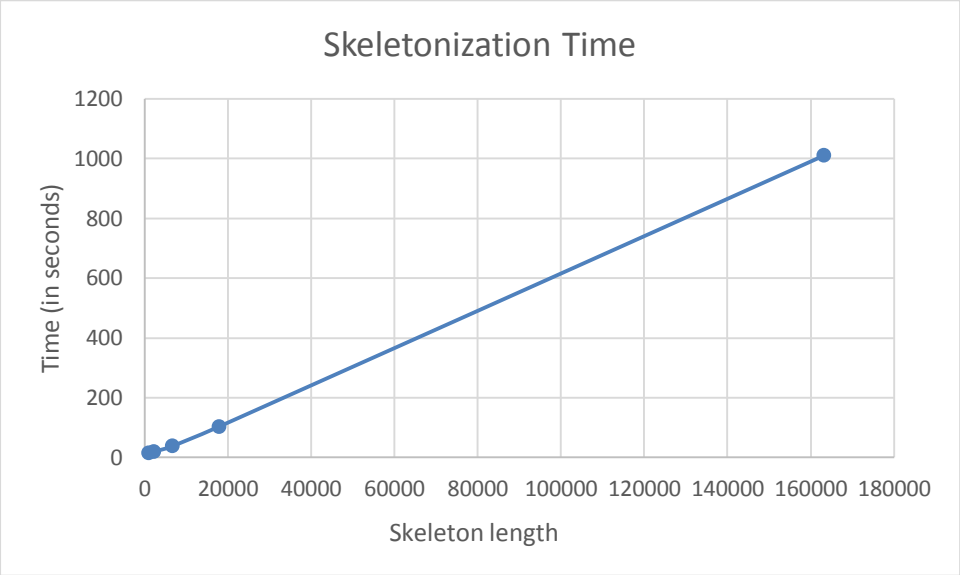


Figure 4.21: Processing time (in seconds) of skeletonization based tracing method. The horizontal axis shows number of voxels in the skeleton and vertical axis shows the time.

Figure 4.21 shows how the processing time increases as the number of voxels in the skeleton increase. The plot shows that our method scales linearly in general and therefore, our method can be used to trace large data sets very quickly.

The processing time in our method includes the time to preprocess the images and the time to generate the skeleton from the preprocessed images. We compared our processing times with Yang's method [13]. In Yang's method, the processing times of vascular tracing were 24.164, 49.291 and 122.599 seconds for the total traced vessel length of 2936, 6756 and 16702 voxels respectively. The experiments were performed on a PC with Intel Core 2 (2.13 GHz) processor under Windows 7 operating system. The configuration is almost similar to my machine. Figure 4.17 shows the comparison.

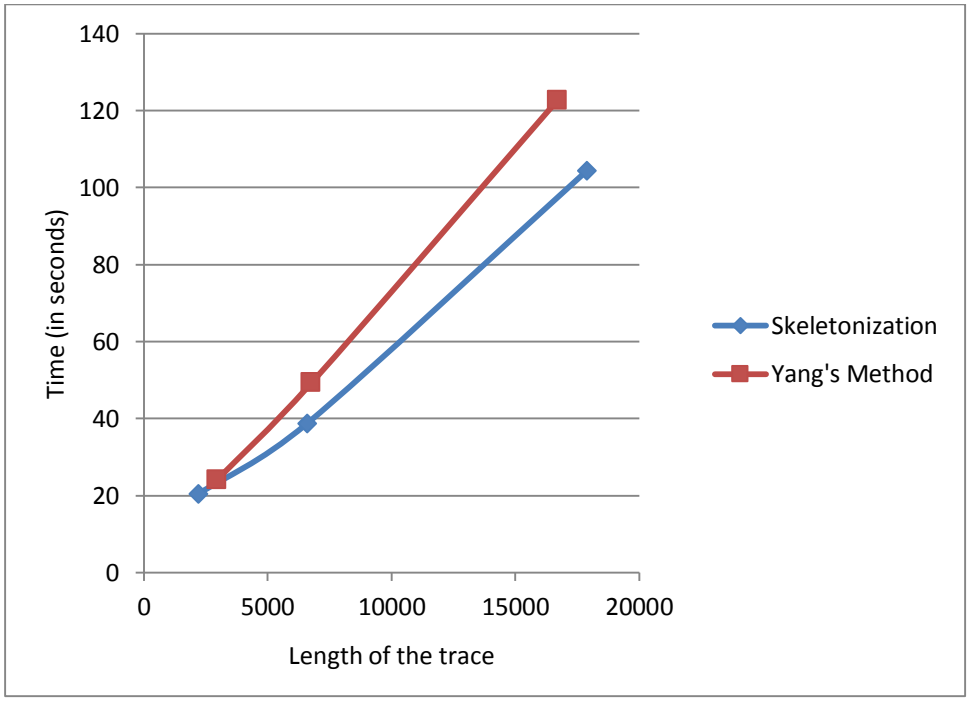


Figure 4.22: Processing time comparison. The horizontal axis shows the size of traced vessel in voxels and vertical axis shows the time in seconds.

Figure 4.22 shows that the skeletonization based tracing method is faster than Yang's method, especially on large data volumes. Yang had compared the processing speed of their method with Han's method [12] and Yang's method is significantly faster than Han's method. In Mayerich's method [11], simulations were run on GPU, so it difficult to compare the speed of our method with Mayerich's method.

5. DISCUSSION

In this thesis, a skeletonization based vascular tracing algorithm was presented that can be used to quickly and accurately trace large volumes of KESM India ink data set. In this chapter, contributions made by my research, open issues, and future works in this area are discussed.

5.1 Contribution

The skeletonization based vascular tracing method presented in this thesis generates an accurate medial line of the vascular network in KESM datasets. This tracing method can be used to obtain statistical information about the network such as the number of nodes and branch end points contained in a volume. The network skeleton can be used to generate a VTK trace of the centerline of the vessels and reconstruct the original volume by dilating the skeleton. It has been shown in this research that this method can fully trace very large volumes of KESM India ink dataset with very high accuracy and the reconstructed models are very similar to the original volume. Processing time is an important criteria for the evaluation of any tracing method since the method can potentially be applied on very large datasets. The method presented here is very fast and the processing time is generally less than the processing time reported in other related work. In summary, our tracing method performs very well on KESM India ink datasets and the method can be applied on other similar datasets as well such as golgi dataset.

5.2 Open issues and future work

In this method, the branch termination errors accounted for approximately 3.5 % of the network. These can be due to two reasons: (1) The noise introduced either because of improper staining or during the imaging. (2) Artifacts present in the skeleton.

To reduce errors due to the first reason, different filtering algorithms can be applied during the preprocessing step and results can be compared. The filtering algorithm should be chosen such that it does not increase the time complexity too much. Additionally, we can consider using different filters for different types of noise present in the dataset such as salt and pepper noise, impulse noise, Gaussian noise, etc. Figure 5.1 gives an illustration of imaging artifact present in the original volume and the corresponding error in the generated skeleton.

To reduce errors due to the second reason, some post processing on the skeleton can be performed such that all branches smaller than a specific threshold are removed from the skeleton. This pruning step would calculate the number of voxels between a node and all the end points connected to that node in the skeleton. This would give the branch size and if this branch size is smaller than the threshold, it is most likely an artifact present in the skeleton and not a true branch in the original dataset. Such branches can be removed from the skeleton before proceeding towards the network analysis.

This method can be used to generate the skeleton on very large datasets. However, when working with very large vessels, we need to process a lot of border

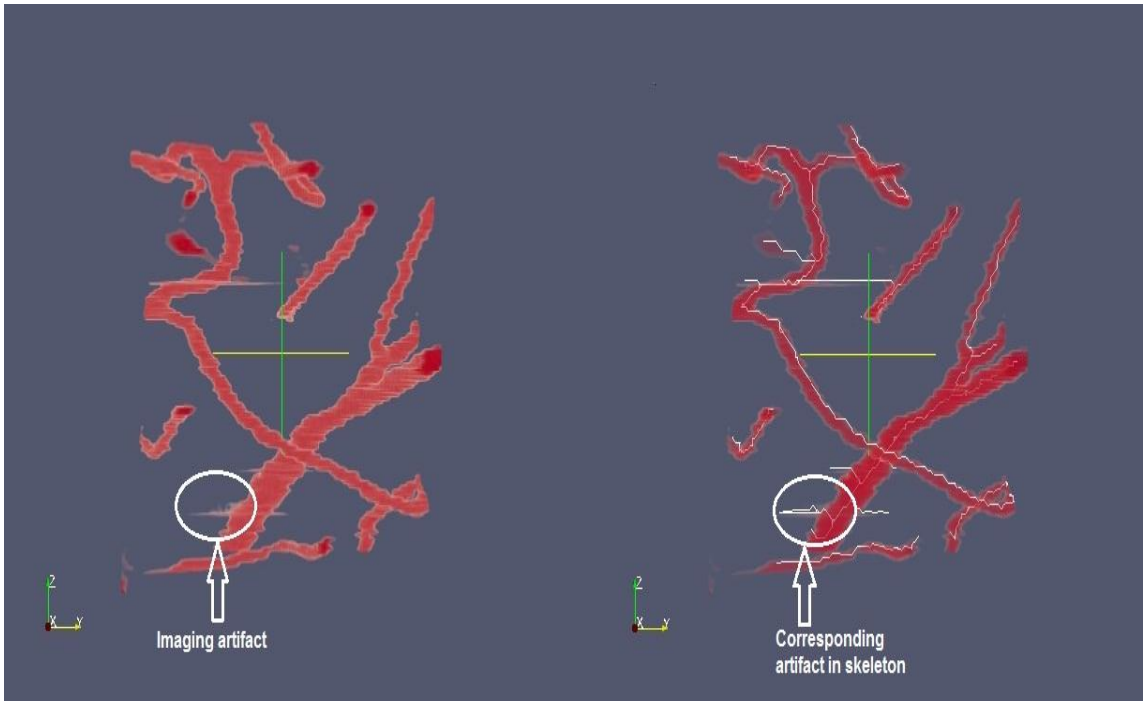


Figure 5.1: Illustration of imaging artifact and the resulting error in the skeleton.

points in parallel. This would increase the memory requirements and to process very large vessels, we would need higher memory.

This method can easily be adapted to perform tracing in golgi dataset in which neuronal morphology is highlighted. However, this dataset contains a lot of gaps. Thinning algorithms have a limitation that they cannot deal with gaps in the datasets. To fully trace such datasets, we need to perform dilation before generating the skeleton. Dilation of the dataset would help in removing these small gaps. Image filtering also would be helpful in removing the image gaps in the dataset.

The skeleton can further be used to generate a graph by considering the branch points and end points as vertices and adding edges between the vertices connected in the skeleton. Once we have generated the graph from the skeleton, we can use various graph algorithms to further analyze the vessels and extract interesting patterns in different regions of the brain.

Finally, future work also can be done on improving the processing time of this method. For this, a large volume and be divided into smaller sub-volumes and the processing can be done in parallel on each sub-volume. Then, the results of different sub-volumes can be combined to generate the resulting skeleton.

6. CONCLUSION

The main goal of this research was to develop a fast and accurate fully automatic method for vascular tracing in KESM mouse brain India ink datasets. This thesis presented a skeletonization based vascular tracing algorithm that can fully trace the vascular network. The proposed method showed high accuracy even in very large datasets. Also, our method is very fast and it took less processing time than the tracing time reported in previous approaches. The skeleton was used to calculate various properties of the dataset such as the number of nodes, the number of branch termination points present, etc. The skeleton was also used to reconstruct the vascular model in the original dataset by dilating the skeleton with appropriate radius. The reconstructed models were both visually analyzed and quantitatively validated for accuracy. These models were very similar to the original datasets and showed high accuracy. This method is expected to help reconstruct and analyze the neurovascular structure in large datasets similar to the KESM dataset.

REFERENCES

[1] J. R. Chung, C. Sung, D. Mayerich, J. Kwon, D. E. Miller, T. Huffman, *et al.*, "Multiscale exploration of mouse brain microstructures using the knife-edge scanning microscope brain atlas," *Frontiers in neuroinformatics*, vol. 5, pp. 1-3, 2011.

[2] B. V. Zlokovic, "The blood-brain barrier in health and chronic neurodegenerative disorders," *Neuron*, vol. 57, pp. 178-201, 2008.

[3] Y. Choe, L. Abbott, D. Han, P. Huang, J. Keyser, J. Kwon, *et al.*, "Knife-edge scanning microscopy: high-throughput imaging and analysis of massive volumes of biological microstructures," In R. Rao and G. Cecchi, editors, *High-Throughput Image Reconstruction and Analysis: Intelligent Microscopy Applications*, Series on Bioinformatics and Biomedical Imaging, Artech House Publishers, pp. 11-37, 2008.

[4] D. Mayerich, L. Abbott, and B. McCormick, "Knife-edge scanning microscopy for imaging and reconstruction of three-dimensional anatomical structures of the mouse brain," *Journal of microscopy*, vol. 231, pp. 134-143, 2008.

[5] J. S. Lim, "Two-dimensional signal and image processing," *Englewood Cliffs, NJ, Prentice Hall, 1990, 710 p.*, vol. 1, 1990.

[6] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, pp. 23-27, 1975.

[7] M. Kerschnitzki, P. Kollmannsberger, M. Burghammer, G. N. Duda, R. Weinkamer, W. Wagermaier, *et al.*, "Architecture of the osteocyte network correlates with bone material quality," *Journal of bone and mineral research*, vol. 28, pp. 1837-1845, 2013.

[8] T.-C. Lee, R. L. Kashyap, and C.-N. Chu, "Building skeleton models via 3-D medial surface axis thinning algorithms," *CVGIP: Graphical Models and Image Processing*, vol. 56, pp. 462-478, 1994.

[9] Brain Networks Laboratory.*Knife-edge scanning microcope(KESM)* (03/31/2015). Available: http://research.cs.tamu.edu/bnl/kesm.html

[10] Y. Choe, D. Mayerich, J. Kwon, D. E. Miller, J. R. Chung, C. Sung, *et al.*, "Knife-edge scanning microscopy for connectomics research," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, 2011, pp. 2258-2265.

[11] D. Mayerich and J. Keyser, "Hardware accelerated segmentation of complex volumetric filament networks," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 15, pp. 670-681, 2009.

[12] D. H. Han, "Rapid three-dimensional tracing of the mouse brain neurovasculature with local maximum intensity projection and moving windows," Doctoral Dissertation, Texas A&M University, College Station, 2009.

[13] W. Yang, "Automated neurovascular tracing and analysis of the knife-edge scanning microscope India ink data set," Master's Thesis, Texas A&M University, College Station, 2014.

[14] A. Dileepkumar, "Semi-automated reconstruction of vascular networks in knife-edge scanning microscope mouse brain data," Master's Thesis, Texas A&M University, College Station, 2014.

[15] K. Palágyi, "A 3D fully parallel surface-thinning algorithm," *Theoretical Computer Science*, vol. 406, pp. 119-135, 2008.

[16] C. Lohou and G. Bertrand, "A 3D 6-subiteration curve thinning algorithm based on P-simple points," *Discrete applied mathematics*, vol. 151, pp. 198-228, 2005.

[17] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker, "Hamilton-jacobi skeletons," *International Journal of Computer Vision*, vol. 48, pp. 215-231, 2002.

[18] N. D. Cornea, D. Silver, X. Yuan, and R. Balasubramanian, "Computing hierarchical curve-skeletons of 3D objects," *The Visual Computer*, vol. 21, pp. 945-955, 2005.

[19] R. C. Gonzalez and R. E. Woods, "Digital image processing, 2nd," *SL: Prentice Hall*, vol. 2, 2002.

[20] S. P. R. Fisher, A. Walker and E. Wolfart. (04/05/2015). *Skeletonization/Medial Axis Transform*. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/skeleton.htm

[21] D. Mayerich, J. Kwon, Y. Choe, L. Abbott, and J. Keyser, "Constructing high resolution microvascular models," in *Third Workshop on Microscopic Image Analysis with Applications in Biology*, 2008.